Available online at www.sciencedirect.com
SCIENCE DIRECT
Image and Vision Computing 24 (2006) 1146–1162
image AND vision COMPUTING
www.elsevier.com/locate/imavis

Extracting 3D information from broadcast soccer video

Yang Liu [a,*], Dawei Liang [a], Qingming Huang [b], Wen Gao [a,b]

[a] School of Computer Science and Technology, Harbin Institute of Technology, No. 92, West Da-Zhi Street, Harbin 150001, China
[b] Graduate School, Chinese Academy of Sciences, Beijing 100039, China

Received 3 October 2005; received in revised form 3 April 2006; accepted 19 April 2006

Abstract

In this paper, we propose a new method to estimate players' and ball's positions from monocular broadcast soccer video. With the relationship between objects and the camera in perspective projection, we derive the formula for estimating the moving objects' positions in real world, even when the ball is in the air. This method calibrates the camera's position in the stadium through the homography between the image and the playfield, and the self-calibration for rotating and zooming camera. Thus, the method can estimate the ball's position in the air without referring to other reference object with known height. In order to reduce manual interference, the players are detected based on the playfield detection. For the ball, we combine the detection procedure and tracking procedure organically. First, we extract candidate regions in each frame, then search the most likely regions in consecutive frames using Viterbi decoding algorithm. Once detected, the ball will be tracked by Kalman filter, which can help improve the detection recall. The system checks whether the ball is lost automatically. If it is lost, the detection procedure restarts. Experiments on synthesized data verify the proposed method, and promising results are obtained on real video data.

Keywords: 3D estimation; Ball detection; Tracking; Soccer video; Broadcast

1. Introduction

In the last decade, researchers pay much attention to estimate players' and ball's positions in real world for field sports for different purposes. In [1,2], the authors estimate players' and/or ball's 3D position in real world to help to understand the game, which is useful for video retrieval. In the meantime, other literatures [3,4] exploit the estimated players' and ball's positions information for 3D reconstruction in order to enrich the video for better enjoyment of the game. Therefore, it inspires much research on estimating players' and ball's 3D information from sports (particularly soccer) video.

Although a lot of methods have been proposed to tackle this problem, algorithms which are based on less assumption and generic video data have to be further investigated. Here, less assumption means that the objects' 3D information is estimated only through the information extracted from sports video without any presupposition of an object's size; while generic video refers to video captured from TV, not from multiple cameras or specific devices, such as high speed camera.

There exist some difficulties to estimate the moving objects' 3D positions, especially for the flying ball in the air, from monocular broadcast soccer video. These difficulties lie in:

(1) We do not know the camera's parameters, including the intrinsic parameter and the extrinsic parameter.
(2) It is nontrivial to extract 3D information from single view, especially for an object in the air.
(3) Needing much human interference.
(4) Ball is so small that it is difficult to detect it in one frame. What's more, it tends to be blurred by camera operation and its rapid speed, etc.

By considering the above issues, in this paper we propose a new algorithm to estimate the ball's 3D information without referring to other object with known height and the required parameters for computation are extracted from video sequence. To further reduce manual labeling, the proposed system detects ball and players automatically. Especially for ball, we present a new scheme to detect it based on Viterbi decoding algorithm by observing consecutive frames.

The system includes the following key steps:

(1) Calibrating the camera extrinsic parameter by self-calibration and the homography between image and the playfield.

* Corresponding author. Tel.: +86 451 86416485.
E-mail addresses: yliu@jdl.ac.cn (Y. Liu), liuy@vilab.hit.edu.cn (Y. Liu).

0262-8856/$ - see front matter © 2006 Elsevier B.V. All rights reserved.
doi:10.1016/j.imavis.2006.04.001

(2) Detecting players and ball in image. The players are detected based on playfield detection and tracked by particle filter based on support vector regression [5]; for the ball, we detect candidates in each frame, then determine the ball in consecutive frames by Viterbi decoding algorithm.
(3) Determining the plane in which the ball flies by finding a most likely parabola.

The rest of the paper is organized as follows. In Section 2, related works are reviewed. Section 3 discusses the geometry relationship between the objects and the camera, and derives the formulas for estimating their 3D positions. In Section 4, we present the methods for ball detection and tracking as well as players detection. Experimental results are shown and analyzed in Section 5. We conclude the paper in Section 6.

2. Related work

In the recent decades, dozens of methods to estimate players' and/or ball's positions in real world (or on playfield) from sports video have been reported. According to the data they process, these research works can be categorized into two classes: the first one focuses on the research work on the data captured by multiple cameras; the second one aims at reconstructing 3D information from monocular video.

The 3D information extraction techniques based on multiple cameras have been applied to sports video analysis and 3D reconstruction, since it is relatively easy. Ren [6] and Xu [7] employ eight stationary cameras placed along the stand to cover the field in the soccer match. In their system, these cameras are calibrated to the ground-plane coordinate system in advance. Thus the players' positions can be determined through the homography between image and the playfield in monocular view. Usually, to estimate the flying ball's position in the air, it needs at least two cameras and using the epipolar geometry restriction. It may work even the ball or players are occluded in some view, since these objects can be seen by other cameras. Iwase et al. [8–10] also adopt multiple cameras to track players and apply the results in free-view visualization generation. In their system, the players are tracked in a virtual ground image instead of in original image. This method can avoid mixing objects close to each other in original image while far away in fact. However, the authors do not describe how to detect and track the ball. With many restrictions, such as using expensive equipment or elaborate calibration, researchers focus their research on broadcast video. Kim [11] generates global scene through mosaic based on the line tracked on the playfield from broadcast video. When two synchronized videos are available, the ball's position can be computed through the multiple view geometry relationship [12]. Bebie and Bieri [13,14] estimate players' positions on playfield, and further estimate the ball's height. They also have to synchronize videos, which is a difficult task for broadcast video. As an application, the results are employed to generate virtual 3D cartoon of a given view point, in which the players' texture are extracted from image. Employing multiple stationary cameras to survey the soccer match can reduce the influence of object occlusion. And the information from multiple cameras makes it possible to estimate whether the ball is flying in the air or not. Although 3D estimation based on multiple views can give good accuracy and robustness [7], it is quite difficult for general users to place so many cameras in stadium in advance. In addition, even if the data is recorded from broadcast videos, two video segments covering the same scene have to be synchronized [11,14].

Compared with the research on multiple view data, other researchers focus their attention on monocular video since it is widely used and easier to obtain than multiple view data, such as recorded from broadcast. Simultaneously, sports video analysis for highlight retrieval is performed on this kind of data usually. In [1,2,4,13,15], the authors compute the tracked players' position on playfield based on homography between image and the ground, which is calibrated by the points of intersection of the lines on the playfield. Ancona and Orazio [16,17] detect ball in image using support vector machine and neural network, respectively, while do not provide its real physical position. As they mount a high-speed camera near the base line in soccer field, the ball image in its view has higher resolution, so texture and structure features can be used, which is lack in the broadcast video. In [18], ball candidates are first obtained in each video frame based on the side product of playfield detection. Afterwards, Kalman filter is employed to generate candidate trajectories from which ball trajectories are selected and extended and in [3] they use the method proposed in [19] to reconstruct 3D scene. As [19] reports, the authors estimate the ball's height with reference to human's height. In their algorithm, they manually determine two objects perpendicular to the ground and calibrate the camera's projection on the ground. Furthermore, the two objects have to have similar scene depth. By exploiting triangle relationship, the ball height can be computed. Reid [20] uses infinite point light source and the ball's shadow to estimate the ball's 3D position. However, it is could hardly detect the shadow by computer in image. Ohno [21] and Yamada [22] introduce dynamic equations to estimate the ball's position, while it is difficult to acquire the ball's speed from image.

Among the above related works, the most similar one to ours is [19]. We all focus on estimating the ball's position from monocular view. Compared with it, the proposed algorithm in this paper has the following characteristics:

(1) The algorithm can estimate the ball's height without referring to other object which is assumed to have a known height.
(2) The algorithm investigates many cases of visual geometry relationship among camera, image, ball and the playfield, which often happen in sports video.
(3) An improved method is proposed to predict the ball's fling plane.
(4) Ball is detected on consecutive frames by Viterbi decoding algorithm, based on the candidate detection result in each frame. The method can overcome the difficulty of blur and distortion in broadcast video.

3. The method for estimating ball's and players' positions

In this section, we present the proposed method in detail. First, we analyze the geometry relationship among the objects, their projective points on the playfield plane (which will be defined as virtual shadow later) and the camera position. Through the geometry relationship, we derive the formula for estimating the moving objects' positions in real world, following which we specify how to compute the parameters required in the estimation process.

3.1. The geometry relationship of objects in perspective projection

In general, the main cameras are placed on fixed positions around the playfield for soccer broadcast in the soccer match, so it is reasonable to mainly consider reconstructing 3D information in the case of the video captured by a rotating and zooming camera. According to the physical restriction, the ball's 3D position can be estimated from some geometry relationship. Fig. 1 shows the relationship among the objects, including ball, ball's virtual shadow on the playfield (regarding the camera as a point light source), camera position and the plane in which the ball flies. Fig. 1a illustrates the case when the ball's height is lower than the camera's position, which often occurs in soccer video. Fig. 1b shows the case when the ball's height is higher than the camera's position, and Fig. 1c shows the case when the ball and camera have the same height relative to the playfield. The proposed method can deal with these cases by a uniform formula. The following subsections interpret the relationship, and derive the formula for computing the ball's 3D position.

- *Coordinate system*. To reconstruct 3D scene, the first key step is to set up coordinate system in the world. Let the playfield plane be the *XOY* plane of the world coordinate system, and the origin be at the center of the base line of the playfield. The axis *X* is perpendicular to the baseline and the axis *Z* is upward.
- *Camera position*. The camera is mounted at a fixed position with its coordinate $\mathbf{t}_{cw}=(X_c,Y_c,Z_c)^{\mathrm{T}}$ in the world coordinate system. Generally, the information is unknown in broadcast video, so it has to be calibrated from the video.
- *Objects' positions*. In principle, objects' positions can be categorized into two classes. The first is on the ground, such as players and ball on the playfield. From the knowledge of computer vision, it is trivial to compute their positions on the playfield from image through the transform between image plane and the playfield. The second is the ball flying in the air. In the second case, Let $\mathbf{p}_u=(X_u,Y_u,0)^{\mathrm{T}}$ and $\mathbf{p}_e=(X_e,X_e,0)^{\mathrm{T}}$ denote the jumping-off point and the falling point of ball, respectively, as shown in Fig. 1. Based on the physical restriction, it is assumed that the ball flies in the plane π, which passes the two points $\mathbf{P}_u$ and $\mathbf{P}_e$, and is perpendicular to the ground. The equation of the plane is described by (1):

$$\pi:\begin{cases} k\cdot X+Y+d=0, k=-(Y_e-Y_u)/(X_e-X_u), \\ \text{if } X_e-X_u\neq 0 \\ Y=-d, \ \text{otherwise} \end{cases} \tag{1}$$

- *Virtual shadow*. The virtual shadow (denoted as $\mathbf{b}_s=(X_s,Y_s,0)^{\mathrm{T}}$) [19] is defined as the intersection between the playfield plane (*XOY*) and the ray which passes through the camera's position $\mathbf{t}_{cw}=(X_c,Y_c,Z_c)^{\mathrm{T}}$ and the ball's position $\mathbf{b}_w$. Here, the line l is written as

$$l:\begin{bmatrix}X\ Y\ Z\end{bmatrix}^{\mathrm{T}}=\begin{bmatrix}X_c-X_s\ Y_c-Y_s\ Z_c\end{bmatrix}^{\mathrm{T}}\cdot t+\begin{bmatrix}X_s\ Y_s\ 0\end{bmatrix}^{\mathrm{T}}, \tag{2}$$

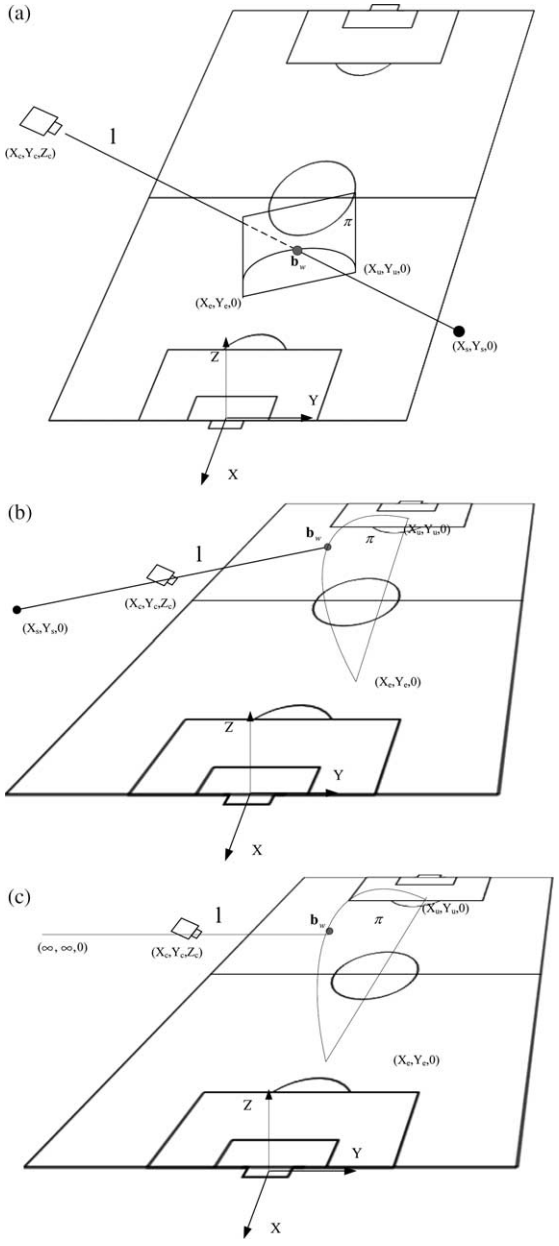

Fig. 1. The geometry relationship among camera position, ball and its virtual shadow: (a) is the case of the ball position is lower than the camera position, (b) is the case of the ball position is higher than the camera position, (c) is the case of the ball and the camera have the same height, in this case the ball's virtual shadow is at infinity.

where t is a free parameter. As the name indicates, the virtual shadow can be regarded as the ball's shadow generated by a light ray emitted from the camera, and the points on the ray have the same virtual shadow. When the ball is moving on the ground, the virtual shadow and the ball are superposition, otherwise they are separate.

- *Computing the ball's position in the air* $\mathbf{b}_w$. Combining Eqs. (1) and (2), we have:

$$\mathbf{b}_w = \begin{bmatrix} X\ Y\ Z \end{bmatrix}^{\mathrm{T}} = \begin{bmatrix} X_c - X_s\ Y_c - Y_s\ Z_c \end{bmatrix}^{\mathrm{T}} \cdot t + \begin{bmatrix} X_s\ Y_s\ 0 \end{bmatrix}^{\mathrm{T}} \tag{3}$$

$$t = \begin{cases} \dfrac{-d - k \cdot X_s - Y_s}{(Y_c - Y_s) + k \cdot (X_c - X_s)}, & \text{if } X_e - X_u \neq 0 \\ \dfrac{X_s}{X_s - X_c}, & \text{otherwise} \end{cases}$$

From Fig. 1, we can obtain the following conclusions: (i) when the ball (or player) is on the ground, if we know the transform between the playfield plane and the image plane, the ball's (or player's) position can be determined; (ii) when the ball is in the air, the ball's 3D position is the point of intersection of line l and plane π; (iii) if line l is parallel to the playfield, i.e. the virtual shadow is at infinity, and the ball's image at the vanish line of the playfield in the image, we can also compute the ball's real position through formula (3), since l will have point of intersection with π; (iv) if line l is always; parallel to the plane π (i.e. the retina is always perpendicular to π), we cannot estimate the ball's position in the air; (v) if the position of the camera is on the field plane, the ball's even the player's position cannot be computed, because this will result in singularity in the transform between image plane and the playfield. It is needed to point out that the latter two cases (iv) and (v) hardly occur in practice. Now, we can see that, to estimate the ball's position in the air it have to estimate its virtual shadow, the camera position and determine the flying plane all from image sequence. The followings will specify these problems in detail.

3.2. Virtual shadow computation

In this section, we describe how to compute the virtual shadow from image.

3.2.1. Camera model

Let $\mathbf{M} = (X,Y,Z)$ be a point in space, with the homogenous coordinate $\tilde{\mathbf{M}} = (X,Y,Z,1)$ and its image is $\mathbf{m} = (u,v)$, whose homogenous coordinate is $\tilde{\mathbf{m}} = (u,v,1)$. According to the pin-hole camera model, the point in space and its image have the following relationship

$$\tilde{\mathbf{m}} \approx \mathbf{K}\begin{bmatrix} \mathbf{R} & \mathbf{t} \end{bmatrix}\tilde{\mathbf{M}} \tag{4}$$

where $\approx$ defines two matrices up to a scale factor. In (4), $\mathbf{K}$, called intrinsic matrix, is a 3×3 matrix with the form of

$$\mathbf{K} = \begin{bmatrix} \alpha & \gamma & u_0 \\ 0 & \beta & v_0 \\ 0 & 0 & 1 \end{bmatrix}, \tag{5}$$

where α and β are the horizontal and vertical focal length, respectively. (u_0,v_0) is the principal point coordinate. γ is the skewness of the two image axes. In order to use linear method to determine $\mathbf{K}$, assuming that $\gamma = 0$, α and β do not vary with the focal length change. In this paper, the principal point is assumed to be at the center of image, and experiments (illustrated later) show that this hypothesis trivially affects the camera position estimation and further the ball's position estimation. $\mathbf{R}$ is a rotation matrix and $\mathbf{t}$ is the coordinates of the origin of the world coordinate system in the camera's coordinate system. They are called the camera's extrinsic parameters.

3.2.2. Homography between image plane and playfield

The playfield plane can be denoted as $Z = 0$ and is substituted into (4), then we have:

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} \approx \mathbf{K}\begin{bmatrix} \mathbf{r}_1 & \mathbf{r}_2 & \mathbf{r}_3 & \mathbf{t} \end{bmatrix}\begin{bmatrix} X \\ Y \\ 0 \\ 1 \end{bmatrix} \approx \mathbf{K}\begin{bmatrix} \mathbf{r}_1 & \mathbf{r}_2 & \mathbf{t} \end{bmatrix}\begin{bmatrix} X \\ Y \\ 1 \end{bmatrix}. \tag{6}$$

Let $\mathbf{M}_p = [X,Y]^{\mathrm{T}}$ denote the point on playfield with its homogenous coordinates $\tilde{\mathbf{M}}_p = [X,Y,1]^{\mathrm{T}}$, then (6) can be depicted in a concise form:

$$\tilde{\mathbf{m}} \approx \mathbf{H}\tilde{\mathbf{M}}_p, \quad \text{where } \mathbf{H} \approx \mathbf{K}\begin{bmatrix} \mathbf{r}_1 & \mathbf{r}_2 & \mathbf{t} \end{bmatrix} \tag{7}$$

3.2.3. Virtual shadow computation using homography

Eq. (7) describes the correspondence relationship between two planes. That is to say, when an image's $\mathbf{H}$ is known and invertible, for a point in image, we can use H's inverse to obtain a point on the field. From (7), we can see that the obtained point, image point and the camera's optic center are on the same line, so the obtained point is the virtual shadow. If H is singular, the virtual shadow cannot be determined, this corresponds to the case (v) in Section 3.1, otherwise it has the following facts. For a point in image: if it is a point on the playfield, through (8), its coordinate in world reference frame can be calculated; if the point is not on the playfield, the acquired vector is the coordinate of the virtual shadow:

$$\tilde{\mathbf{M}}_p \approx \mathbf{H}^{-1}\tilde{\mathbf{m}} \tag{8}$$

3.3. Computing homography

The 3×3 matrix $\mathbf{H}$ is called homography matrix. If the matrix is invertible, it has eight independent components. So it needs four corresponding points to determine $\mathbf{H}$. Fig. 2 shows the soccer field model. As [23] regulates, the size of the field is

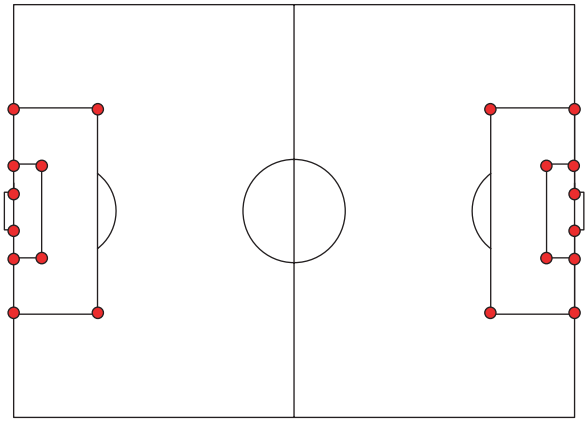

Fig. 2. The playfield model in soccer match. As the width and length of it are not specified in FIFA's law, then only red points in the figure can be used to compute the homography between image and the playfield. If we want to compute the homography at kick-off circle, it has to estimate the size of the field.

not unique, thus only the circle points in the figure can be used to compute the homography matrix. When an image has enough of these points, $\mathbf{H}$ can be computed directly. For the image with insufficient corresponding points, the image's $\mathbf{H}$ is calculated indirectly. Because the camera is mounted at a fixed position, the image points $\tilde{\mathbf{m}}_{t-1}$ and $\tilde{\mathbf{m}}_t$ of a still point $\tilde{\mathbf{M}}$ in space in two adjacent frames have the transform

$$\tilde{\mathbf{m}}_t \approx \mathbf{P}_{t,t-1}\tilde{\mathbf{m}}_{t-1}, \tag{9}$$

where $\mathbf{P}_{t,t-1}$ has the same property with $\mathbf{H}$. In some literatures, $\mathbf{P}_{t,t-1}$ is called inter-frame homography. To differentiate it from $\mathbf{H}$, it is called global motion parameter in this paper. The derivation of (9) is showed in the appendix. Let $\mathbf{H}_{t-1}$ and $\mathbf{H}_t$ be the homography matrices of frame $t-1$ and frame t, respectively. According to (7), we have

$$\begin{cases} \tilde{\mathbf{m}}_{t-1} \approx \mathbf{H}_{t-1}\tilde{\mathbf{M}} \\ \tilde{\mathbf{m}}_t \approx \mathbf{H}_t\tilde{\mathbf{M}} \end{cases} \tag{10}$$

Substituting (9) into (10), the following recursive function is acquired

$$\mathbf{H}_t \approx \mathbf{P}_{t-1,t}\mathbf{H}_{t-1} \approx \mathbf{P}_{t-1,t}\mathbf{P}_{t-2,t-1}\mathbf{H}_{t-2} \approx \cdots \approx \mathbf{P}_{t-1,t} \cdots \mathbf{P}_{t-k,t-k+1}\mathbf{P}_{t-k} \tag{11}$$

Eq. (11) tells us that if some image's homography matrix in a video sequence is known, then the $\mathbf{H}$ matrix of image with insufficient corresponding points can be estimated based on (11). Kim [19] also uses the results, where the authors mosaic the whole scene of the sequence. For the first frame, the control points are labeled by hand. In the future, we can exploit the method in [2].

3.4. Calibrating the camera position

Camera position is another important factor for estimating the ball's 3D position. Let us study the relationship between $\mathbf{H}$ and the intrinsic and extrinsic parameters in (7), then we have

$$\mathbf{K}^{-1}\begin{bmatrix}\mathbf{h}_1 & \mathbf{h}_2 & \mathbf{h}_3\end{bmatrix} \approx \begin{bmatrix}\mathbf{r}_1 & \mathbf{r}_2 & \mathbf{t}\end{bmatrix}, \tag{12}$$

where $\mathbf{h}_i$, $i=1...3$ is a column of $\mathbf{H}$. According to (12), the camera's extrinsic parameters are calculated by (13)

$$\mathbf{t} = s\cdot\mathbf{K}^{-1}\mathbf{h}_3; \qquad \mathbf{r}_1 = s\cdot\mathbf{K}^{-1}\mathbf{h}_1; \qquad \mathbf{r}_2 = s\cdot\mathbf{K}^{-1}\mathbf{h}_2 \tag{13}$$

where $s=1/\|\mathbf{K}^{-1}\mathbf{h}_i\|$, $i=1,2$ and $\mathbf{r}_3=\mathbf{r}_1\times\mathbf{r}_2$ [24] using the restriction of $\mathbf{R}$ being an orthonormal matrix. Here, $\mathbf{t}$ is the coordinate of the origin world coordinates system in the camera's coordinate system, then the camera position in the world is:

$$\mathbf{t}_{cw} = -\mathbf{R}^{-1}\mathbf{t} \tag{14}$$

It is seen from (13) and (14), in order to compute $\mathbf{t}_{cw}$, an image's $\mathbf{H}$ and the intrinsic parameter $\mathbf{K}$ when capturing the image are needed to be known in advance. Since the camera adopted in soccer broadcast can be regarded as rotating and zooming camera, we adopt the method proposed in [25]. The principal point is assumed to be at the center of image (as literature [26] and our experiments show that this setting hardly affects the result), then the components α and β of $\mathbf{K}$ can be acquired by solving equation system (15)

$$\begin{bmatrix} p_{11}p_{21} & p_{12}p_{22} \\ p_{11}p_{31} & p_{12}p_{32} \\ p_{11}p_{31} & p_{22}p_{32} \end{bmatrix} \begin{bmatrix} \alpha^2 \\ \beta^2 \end{bmatrix} = \begin{bmatrix} -p_{13}p_{23} \\ -p_{13}p_{33} \\ -p_{23}p_{33} \end{bmatrix}, \tag{15}$$

where p_{ij}, $i,j=1...3$ is the component of the global motion parameter $\mathbf{P}$.

3.5. The flying plane determination

To determine the flying plane function, it has to find two points which the ball flies through. The jumping-off point and the falling point are labeled by hand, as it is too difficult to do it fully automatically.

However, in some cases, the ball is kicked again before it reaches the playfield, thus the falling point of the ball cannot be determined, so we have to estimate the ball's flying plane by other method given the jumping-off point. From the knowledge of kinetics, if air resistance is neglected, the ball will fly along a parabola. In [19], the authors regard the plane with the minimal parabola fitting error as the flying plane. In our experiment, we find out that in some cases only fitting parabola may fail to predict the plane and Fig. 12a shows an example. As the figure depicts that if the objective function to optimize is only to find the plane with minimal fitting error between the data points and fitting curve, then these data points may not spread evenly. This is obviously unreasonable, because the horizontal distance between two adjacent positions of ball is quite different. Fig. 3 shows an illustration for the case. If the plane, in which only the fitting errors is minimal, is regarded as the flying plane, plane A or plane C may become the flying plane, since they may have the minimal fitting error. When the air resistance is neglected, the flying ball is only affected by the gravity, the horizontal displacements are identical in the orthogonal completion space of acceleration, so we add another physical restriction to the

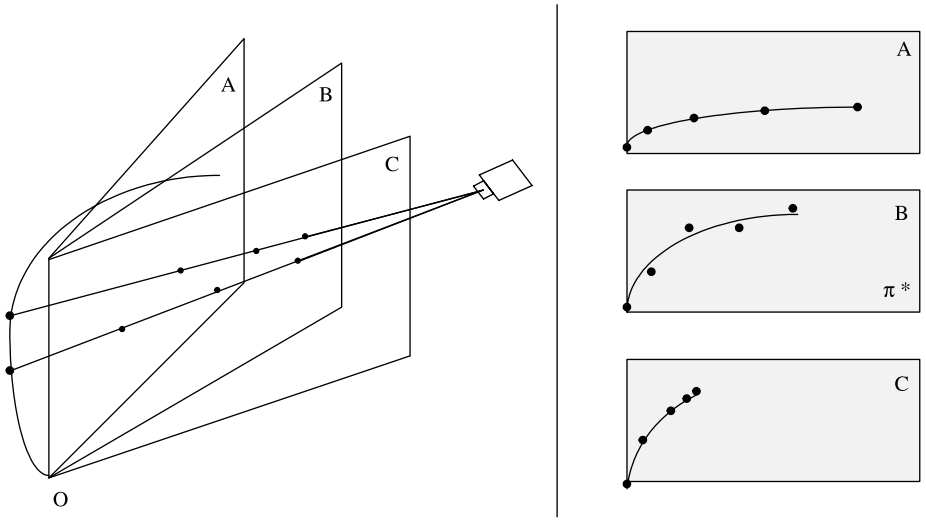

Fig. 3. Illustration of the optimal parabola in different supposing flying plane. The right column shows the optimal fitting cures in different plane. In some cases, planes A or C may have less fitting error than the true optimal plane B (π^*). Thus, the flying plane prediction may fail, if only optimal parabola fitting error is considered.

optimization objective, then the optimization function is written as (16)

$$\pi^* = \arg\min_{\pi} \left(error_{fit}(\pi) + dis_{var}(\pi)\right) \tag{16}$$

where

$$\mathrm{error}_{\mathrm{fit}}(\pi) = \frac{1}{\sum_{i=1}^{N} f_\pi^*\left(x_i^\pi\right)} \sum_{i=1}^{N} \left(f_\pi^*\left(x_i^\pi\right) - z_i^\pi\right)^2, \tag{17}$$

and:

$$\mathrm{dis}_{\mathrm{var}}(\pi) = \frac{1}{\sum_{i}^{N-1} d_i^\pi} \mathrm{var}\left(\{d_1^\pi,\ldots,d_{N-1}^\pi\}\right), \tag{18}$$

$$d_i^\pi = |x_{i+1}^\pi - x_i^\pi|, \quad i = 1,\ldots,N-1$$

In (17), $f_\pi^*(x) = ax^2 + bx + c$ is the optimal fitting parabola for data set $\{(x_i^\pi, z_i^\pi) | i = 1,\ldots,N\}$, which is calculated through (3). In (17) and (18), the denominators are used to normalize the two errors, respectively. Note, here x is the horizontal coordinates of the flying plane, and z is the vertical coordinates of the plane which is equal to the vertical coordinates in the world coordinate system.

4. Moving object detection and tracking

We have shown how to estimate the objects' real positions, especially for the flying ball, in theory. This section describes the methods for detecting them in image. In the first part of this section, a scheme is proposed for ball detection and tracking; then presents the player detection based on automatic background subtraction. At last, we briefly introduce the global motion estimation (GME) employed in our system.

4.1. Ball detection and tracking

The proposed scheme combines ball detection and tracking procedures organically and its flowchart is shown in Fig. 4. In ball detection procedure, color, shape and size are used to extract ball candidates in each frame. Then Viterbi decoding algorithm is applied to extract the optimal path which is most likely to be ball's path in consecutive frames. Once the ball is detected, the tracking procedure based on Kalman filter and template matching is started, which is initialized using the detection results. In each tracking step, ball location is verified to update the template and to guide possible ball re-detection.

4.1.1. Ball detection

4.1.1.1. Ball candidates detection in a frame. Based on the observation that the ball's color is nearly white in long view shots, white pixels are first segmented according to (19) in normalized RGB color space

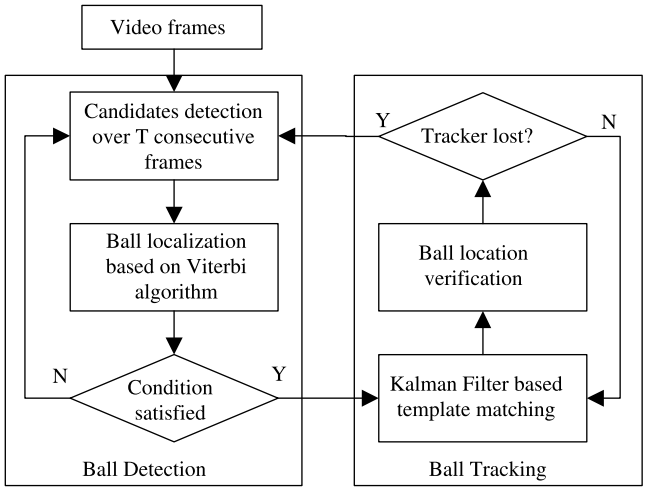

Fig. 4. The flowchart of ball detection and tracking.

$$B(x,y)=\begin{cases}1, & (r(x,y)-1/3)^2+(b(x,y)-1/3)^2\le a^2 \wedge I(x,y)\ge b\\ 0, & \text{otherwise}\end{cases}, \tag{19}$$

here B is a binary image and (x,y) is the pixel location, $r(x,y)$ and $b(x,y)$ denote normalized red and blue component, respectively, $I(x,y)$ denotes luminance value. The thresholds are set to $a=0.05$ and $b=160$ empirically. Morphological close operation is used to eliminate noises, after that a new region growing algorithm proposed in [27] is employed to connect pixels into regions and smooth the boundaries. To obtain ball candidates, several features are used, including the size of the object, the ratio of the length and the width of the object's minimal bounding rectangle (MBR), and the area ratio of the object and its MBR. In order to adapt to various ball appearances, the thresholds are set as loosely as possible to ensure that the true ball region is included in. In the experiment, the threshold of the fist feature is set differently as the object appears at different image position, the threshold of the second one is set to 1.5, and the threshold of the third one is set to 0.5.

4.1.1.2. Graph construction. After candidate detection in T consecutive frames, a weighted graph is constructed. Each graph node represents a ball candidate. Since the ball's locations in two adjacent frames are cose to each other, only those candidate pairs (between adjacent frames) whose Euclidean distance (in image plane) is smaller than the threshold $d_{\max}$ ($=20$ pixels in experiment) contribute to the graph edge set. According to (20) each node is assigned a weight representing how it resembles a ball. Meanwhile each edge is assigned a weight through (22) to represent how likely the two nodes correspond to the same object

$$v_i^t=\begin{cases}1-\sqrt{c_i^t} & c_i^t\le 1\\ 0 & c_i^t>1\end{cases} \tag{20}$$

where

$$c_i^t=\frac{1}{M\mu_r^2}\sum_k(||p_k-\mu||-\mu_r)^2 \tag{21}$$

$$e_{i,j}^t=\frac{\left(\varpi_s s_{i,j}^t+\varpi_g g_{i,j}^t\right)}{\sqrt{1+\left(d_{i,j}^t/d_{\max}\right)^2}} \tag{22}$$

In (20)–(22), superscript t denotes the relative serial number of frame; subscripts i and j denote the ith candidate in frame t and the jth candidate in frame $t+1$, respectively. In (21), c_i^t is called Circular Variance (CV) [28]. The less the CV is, the more the contour resembles a circle. p_k is a contour point, M is the number of contour point, μ is the centroid of the contour, and μ_r is the average distance from contour points to the centroid. In (22), $s_{i,j}^t$ and $g_{i,j}^t$ are the size and the gray level similarity of two candidates, respectively, with ϖ_s and ϖ_g being the corresponding weights. For simplicity we set $\varpi_s=\varpi_g=0.5$. $d_{i,j}^t$ is the Euclidean distance (in image plane) of two candidates.

We assume that $(\Delta w,\Delta h)$ obeys Gaussian distribution, where Δw is the width difference between MBRs of two candidates, and Δh is the height difference between MBRs of two candidates. Therefore, $s_{i,j}^t$ can be defined as (23), where Σ can be estimated from ball samples. In (24), $g_{i,j}^t$ is the gray level normalized cross correlation of two candidate regions, where vectors I_1 and I_2 are obtained through raster scanning of the candidate regions. If the candidate regions are not equal in size, they are adjusted to equal size before scanning:

$$S_{i,j}^t=N(0,\Sigma) \tag{23}$$

$$g_{i,j}^t=\frac{\sum_k I_1(k)\cdot I_2(k)}{\sqrt{\sum_k I_1(k)\cdot I_1(k)}\sqrt{\sum_k I_2(k)\cdot I_2(k)}} \tag{24}$$

4.1.1.3. Ball's path extraction. Finding the optimal path of a graph is a typical dynamic programming problem. Viterbi algorithm is employed to extract it based on the constructed graph. Note that the graph can be constructed incrementally. Viterbi algorithm is described in Algorithm 1 similar to that in [29].

Let P_j^t be the optimal path ending up at jth candidate in frame t, then the notations in Algorithm 1 are explained as follows. N_t is the number of candidates in frame t, δ_j^t is the sum of node and edge weights along P_j^t, $\Psi_t(j)$ is the index linking to the candidate in frame $t-1$ on P_j^t, and $\{q_t\}_{t=1,\ldots,T}$ is the optimal path. If the number of candidates on the optimal path is less than T, the observation window is moved forward by one frame and then the ball detection procedure is performed again. An illustration of the graph and its optimal path is shown in Fig. 5.

Algorithm 1. Ball localization based on Viterbi algorithm

1. Initialization: $\delta_i^1=v_i^1, \quad \psi_1(i)=0, \quad 1\le i\le N_1;$
2. Recursion:
   $\delta_j^t=\max_{1\le i\le N_{t-1},\ d_{i,j}^{t-1}\le d_{\max}}\left(\delta_i^{t-1}+e_{i,j}^{t-1}+v_j^t\right),$
   $\psi_t(j)=\arg\max_{1\le i\le N_{t-1},\ d_{i,j}^{t-1}\le d_{\max}}\left(\delta_i^{t-1}+e_{i,j}^{t-1}+v_j^t\right),,$
   $1\le j\le N_t,\ 2\le t\le T;$

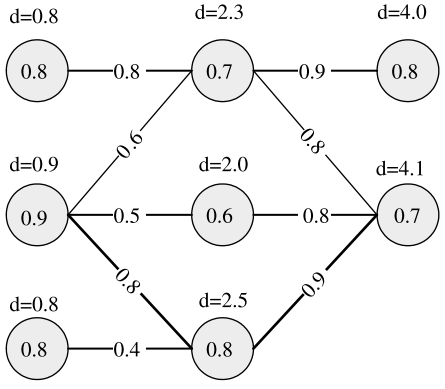

Fig. 5. Illustration of the weighted graph. The optimal path is marked with bold line.

3. Termination: $q_T = \arg\max_{1\le i\le N_T}\left(\delta_i^T\right)$;
4. Path backtracking: $q_t = \psi_{t+1}(q_{t+1})$, $t = T-1, T-2,\ldots,1$.

4.1.2. Ball tracking

Once the ball is detected, Kalman filter-based template matching (in terms of gray level normalized cross correlation) is exploited to track it. Kalman filter predicts the ball's location in the next frame and filters the tracking result in the current frame. Template matching is used to obtain observation. Kalman filter and the template are initialized using detection results.

Kalman filter addresses the general problem of estimating the state X of a discrete time process that is governed by a linear stochastic difference equation

$$X_{k+1} = AX_k + w_k \tag{25}$$

and the measurement Z is:

$$Z_k = HX_k + v_k \tag{26}$$

The random variables w_k and v_k represent the process and measurement noise, respectively. They are assumed to be independent of each other and have normal distribution. In this paper, first order dynamics model is employed, i.e.

$$X = \begin{bmatrix} x \\ y \\ \dot{x} \\ \dot{y} \end{bmatrix}; \qquad Z = \begin{bmatrix} x \\ y \end{bmatrix}; \qquad A = \begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}; \tag{27}$$

$$H = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}.$$

here (x,y) denotes the ball's center, and (x,y) denotes the ball's velocity. Due to lack of space, more details about Kalman filter can be referred to [30]. As the matrix A indicates, we assume that the ball moves with constant velocity on the image plane. Although the assumption is not exactly right, the model can work well in our system.

A simple but effective method is adopted to make the tracker adaptable to the ball's scale change over frames. A slightly larger block $(x1-\Delta,\ y1-\Delta,\ x2+\Delta,\ y2+\Delta)$ is generated for matched ball region $(x1,y1,x2,y2)$, where $(x1,y1)$ and $(x2,y2)$ are the top-left coordinates and the bottom-right coordinates of the matching region, respectively. The same method in Section 4.1.1 is used to extract object, and we employ (20) to evaluate whether it is a ball. If a ball is detected, the template is updated. The number of consecutive missing detections is counted, and if it is larger than a predefined threshold (e.g. 5), the ball detection procedure will restart.

The reason of adopting tracking algorithm instead of only using detection method is that the tracking algorithm will work when the ball region cannot be well segmented, i.e. the tracker will find the ball in next frames, while detection have to verify the ball region in consecutive T (=5) frames. This is some more strict.

4.2. Players detection

This section describes player detection in image. The paper does not comprise player tracking, we refer the readers to [5], where we use improved particle filter to track players, which can help to solve the problem of particle degeneration.

As our project only uses middle and long view images for 3D reconstruction for soccer video, and in this case the players' regions in image are surrounded by playfield region. We segment players in image based on playfield detection.

4.2.1. Playfield detection using GMM

It is observed that in a soccer sequence only some small regions (bins) of a histogram (C_bC_r color space) have non-zero values, and in general there are some peaks in the histogram. Although usually the region with the main peak corresponds to grass color, exceptions could be found. Thus, we have to determine the main region from histogram, which corresponds to the playfield color in the video sequence. The procedure is shown in Algorithm 2. Notice that only the connection region with larger sum of bins in histogram is considered as the playfield color, and this avoids the case that the color with isolated bin with the largest value is regarded as the playfield color which results from video coding in general.

Algorithm 2. Playfield color region detection in the histogram

1. Determine the main peak P_1;
2. Find the connected region (4-connected region) around P_1, only the bins with values larger than T*Value(P_1) are considered. Compute the sum of the connected bins, noted as Sum_1, then subtract the connected region, here T is a ratio. In this paper, we set it 0.05.
3. Similar to 1 and 2, find the main peak P_2 in the remaining region of the histogram and compute the sum of the connected bins around it, denoted as Sum_2.
4. Return the connected region in the histogram corresponding to the larger of Sum_1 and Sum_2.

After the rough distribution region detection in C_bC_r space, Gaussian mixture model is exploited to model the playfield color, which is described in (28)

$$G = \sum_{i=1}^{k} \pi_i G_i, \tag{28}$$

where

$$G_i(X;\theta_i) = \frac{1}{(2\pi)^{d/2}|\Sigma_i|^{1/2}} \exp^{-1/2(X-\mu_i)^{\mathrm{T}}(\Sigma_i)^{-1}(X-\mu_i)}, \quad \text{and}$$

$$\sum_{i=1}^{k} \pi_i = 1$$

Fig. 6. The results of players segmenting are used to initialize the tracker for players. The left column are original images, and the right column are the segmented players.

Each component G_i is a Gaussian function, parameterized by θ_i, which consists of the mean vector μ_i, and the covariance matrix Σ_i. The dimension of sample data X is d. Thus, the set $\{\pi_i,\theta_i\}$ of all unknown parameters belongs to some parameter space. Generally, these parameters are estimated by expectation maximization (EM) algorithm. In our system, we set the model to have two components.

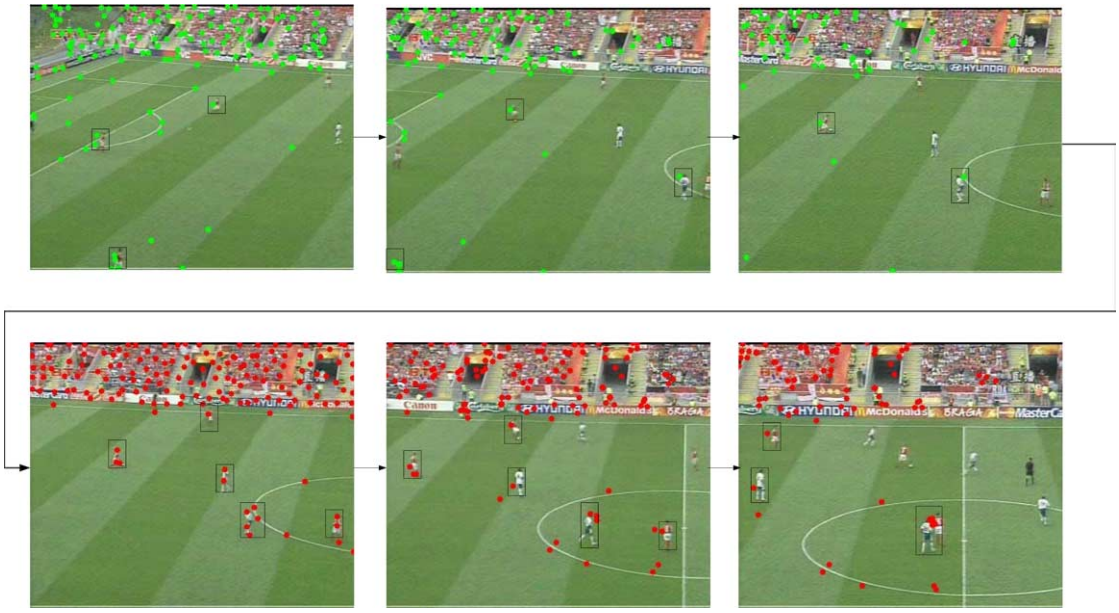

Fig. 7. The extracted and tracked feature points. When the number of feature points is less than a threshold, then restart to extract feature points. In the figure, the color change illustrates the restart process. The points in black rectangle are deleted from feature points list based on the players detection result.

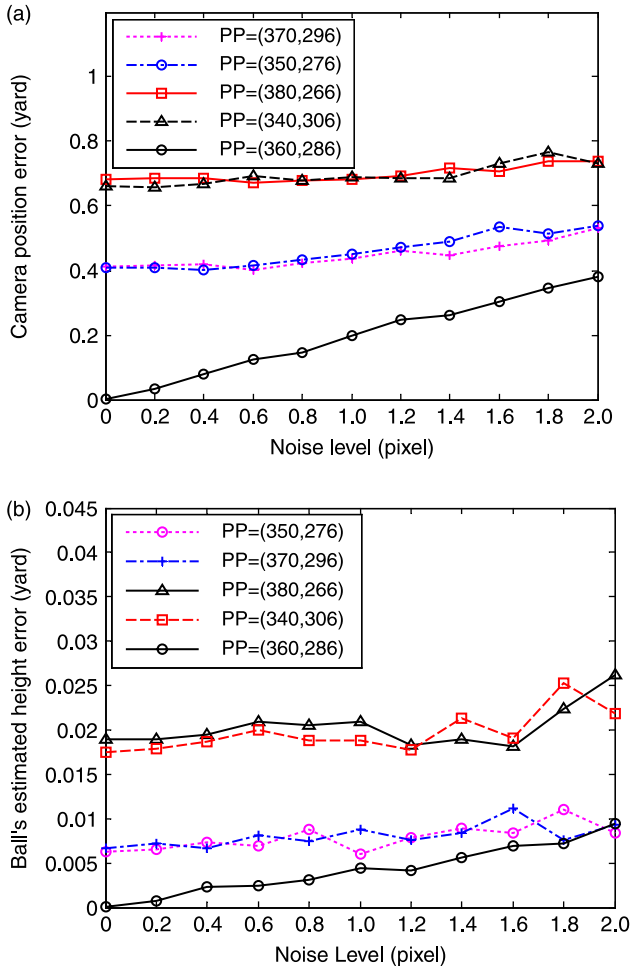

Fig. 8. The effect on camera position (a) and ball position estimation (b) of deviation of principal point position and noise.

4.2.2. Players segmentation

As the result of playfield detection, a binary image is outputted for each image in the sequence, in which 1 denotes playfield pixel and 0 non-playfield pixel. Usually, the regions of players are marked by the binary image, and to obtain better detection result, region-growing procedure is used which is a general technique for image segmentation. We also use the same algorithm as extracting ball candidates to perform the segmentation [27]. After the playfield detection and player segmentation, regions with label 0 and surrounded by region with label 1 are regarded as players. Fig. 6 shows the result. Since the players are on or near the ground, their positions can be estimated by (8).

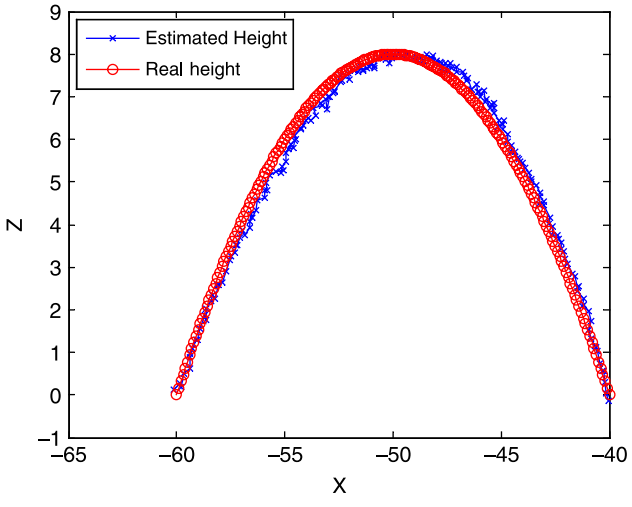

Fig. 9. Ball's height estimation under the condition of its height is higher than the camera position.

Table 1
Ball detection and tracking experiment results

Video sequences	Frame number	Ground truth	Experimental results			
		#Ball	No.of detected and tracked	No.of false positive	Precision (%)	Recall (%)
Sequence1 (bad playfield)	1–100	88	45/67	0/2	100/97.0	51.1/73.9
	101–200	92	25/53	0/0	100/100	27.2/57.6
	201–300	99	88/97	0/10	100/89.7	88.9/87.9
	301–400	97	72/97	0/3	100/96.9	74.2/96.9
	401–500	80	47/62	10/20	78.7/67.7	46.3/52.5
	501–600	96	53/92	5/5	90.6/94.6	50.0/90.6
	601–650	48	25/45	5/13	80.0/71.1	41.7/66.7
	Sum	600	355/513	20/53	94.4/89.7	55.7/76.7
Sequence2 (good playfield)	1–100	92	38/93	0/8	100/91.4	41.3/92.4
	101–200	100	67/99	0/3	100/97.0	67.0/96
	201–300	94	70/95	15/11	78.6/88.4	58.5/89.4
	301–400	81	55/71	10/23	81.8/67.6	55.6/59.3
	401–500	64	27/39	2/3	92.6/92.3	39.1/56.2
	501–600	82	68/84	3/10	95.6/88.1	79.3/90.2
	601–700	99	86/99	0/4	100/96.0	86.9/96.0
	701–719	19	10/18	1/3	90.0/83.3	47.4/78.9
	Sum	631	421/598	31/65	92.6/89.1	61.8/84.5

In the results columns, the number before slash is the result of only using the ball detection procedure, and the number behind it is the result of combining the ball detection and tracking procedure.

4.3. Global motion estimation

To estimate the global motion caused by camera rotating and zooming, we exploit the method of LKT (Lucas–Kanade–Tomasi) good features to track [31] and implemented by OpenCV [32]. First, we extract good feature points from an image. Then the algorithm finds their corresponding points in the next image. Let $\{(x_i^{t-1},y_i^{t-1})|i=1,\ldots,N\}$ and $\{(x_i^t,y_i^t)|i=1,\ldots,N\}$ denote the feature point sets of frames $t-1$ and t, respectively, where x and y are the feature points' horizontal and vertical coordinates in frame, respectively, and N is the number of feature points that have been tracked. In general, these feature points locate in the regions of auditorium, players and the edge arising from occlusion. The later points will move with the players.

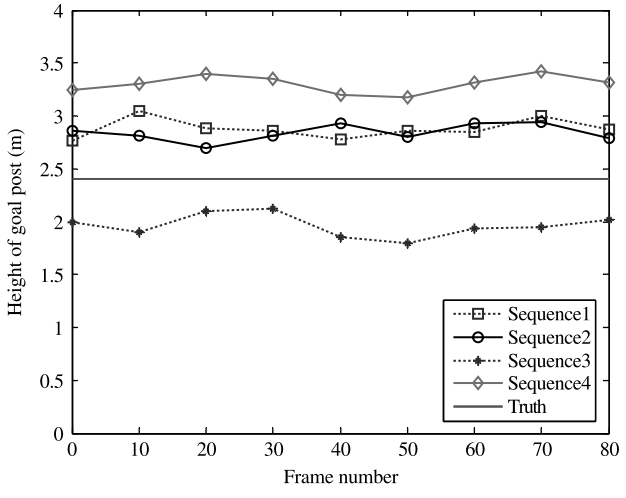

Fig. 10. The estimated goal post height on four sequences. The red line is the true height of the goalpost.

Because the camera has large motion, some of these feature points will be lost in a few seconds. If the number of the tracked points is less than $T*N_0$, the algorithm will restart the feature points detection process and the points tracking algorithm will perform in the remaining images in the sequence, where T is a scalar factor (say $T=0.5$) and N_0 is the number of extracted feature points in the initial image. In Fig. 7, the bottom row illustrates this process, and the feature points are depicted in different colors. As we have pointed out, some feature points are in the region of players and the edge of occlusion between players and the playfield, they will affect global motion parameters estimation. To eliminate the effect of these points on GME, they are removed based on the player detection result in the above section. In Fig. 7, these feature points are surrounded by rectangle. At last, we use the method in [12] to compute the global motion parameters.

5. Experiments and application

In this section, the proposed algorithm is verified and tested on synthesized data and real video data, respectively. At last, the algorithm is applied to extract 3D position information of the moving object from a highlight segment, which is used to generate 3D cartoon.

5.1. Synthesized data

We generate synthesized data in a virtual stadium to verify the algorithm, and study the effects of supposing the principal point located at image center and the occurrence of noise.

According to the law of FIFA [23], we let the virtual playfield be of 100 yards long and 70 yards wide. The world coordinate system is set up as shown in Fig. 1. The virtual camera is mounted at $(-50, -50, 10)$, and the initial focal

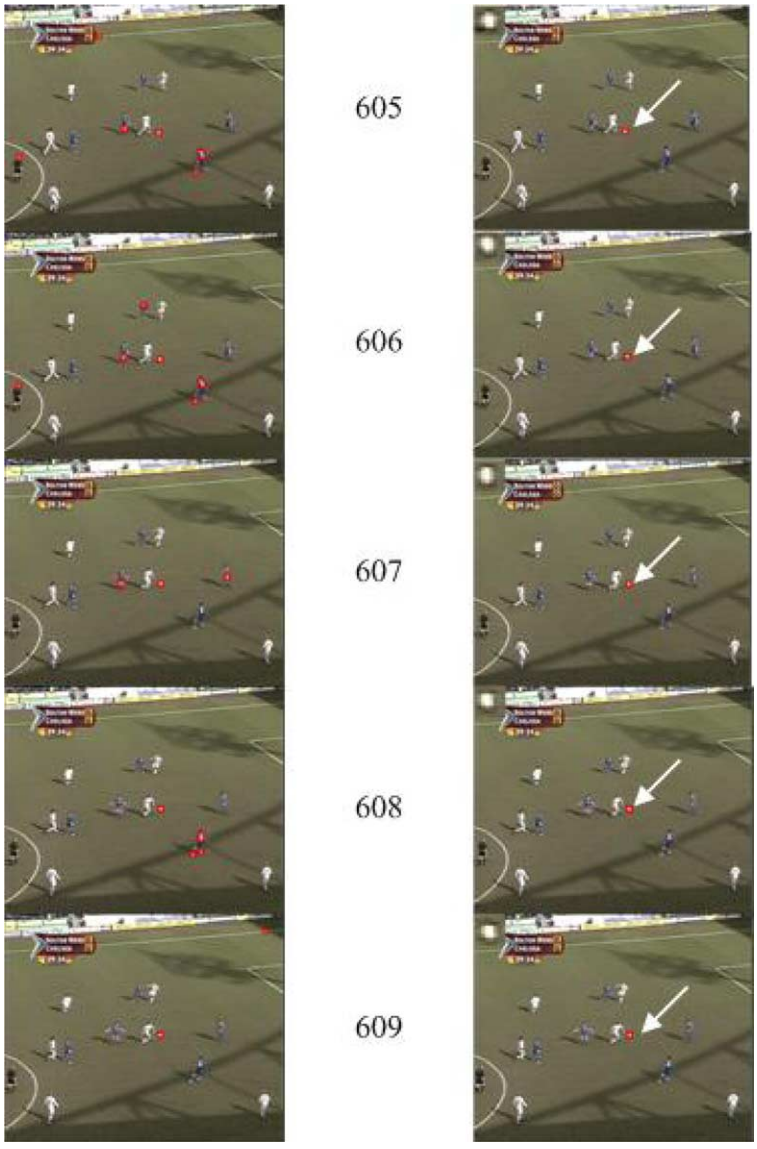

Fig. 11. Illustration of ball detection in consecutive frames. In the left column, ball candidates are marked with red rectangle. Extracted ball locations using Viterbi algorithm are shown in the right column with a white arrow indicating the ball's location in each image. The ball is shown on the top left corner of each image. Middle column shows the frame number.

length is $\alpha = 1200$, $\beta = 1100$. The size of image is 720×572. The camera simulates the action of real video in broadcast video, which pans from right to left and the focal length increases 1% per frame. At the same time, random movement is added to tilt angle. The ball flies from $(-60, 0, 0)$ to $(-40, 0, 0)$ and the highest point is 8 yards. Here, the unit of the world coordinate system is yard.

First, we consider how the degree of principal point position deviating from the center of image and the occurrence of noise affect the camera position and ball's 3D position estimation. The noise obeys normal distribution with mean 0 and standard deviation σ. Fig. 8a illustrates the error between the estimated camera position and the truth, and Fig. 8b shows the error of the ball's estimated 3D position from the true position. At every noise level (11 levels, σ increases from 0 to 2 pixels uniformly), 100 runs are done. In the figures, PP denotes the principal point's real position in the virtual data generation process, while the principal point is always assumed to be at the center of image in the calibration process. From the figures, it is concluded that the estimation precision decreases when the deviation of the real principal point position from the image center increases. However, the effect is trivial (this is consistent with the report in [26]). So it is acceptable to set the center of image as principal point position in practice. Seeing along horizontal, we can study the effect of noise, and notice that the error also increases as σ becomes large. In real application, except for these two factors influence, lens distortion and camera displacement, etc. all will affect the estimation precision to much extent.

As prior section points out, the formula for estimating ball's position can also deal with the case of the ball's position being higher than the camera position. Now, we move the camera to

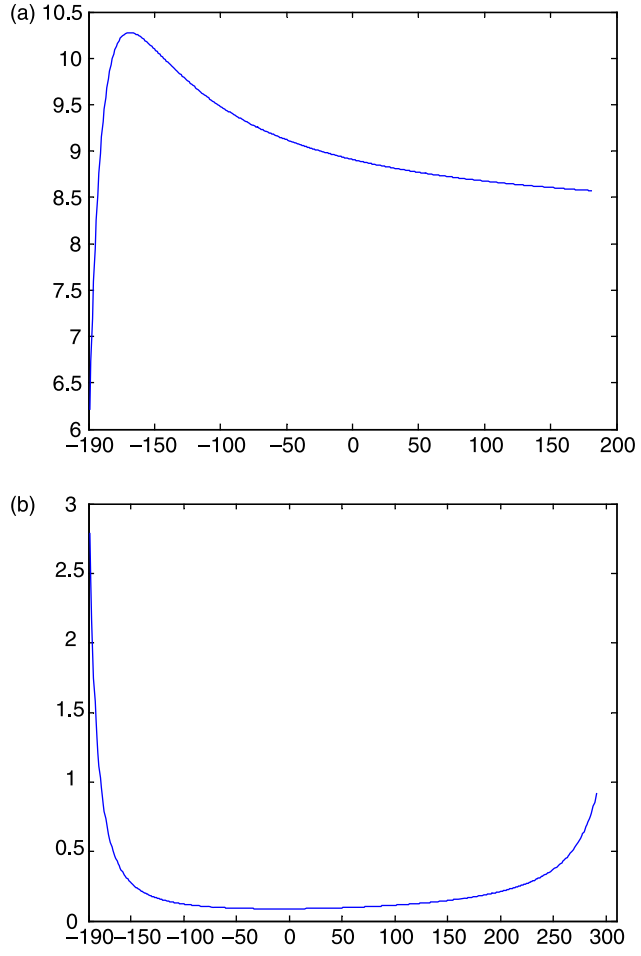

Fig. 12. The obtained optimization function values on different candidate flying plane: (a) is the result by optimizing the parabola fitting error; (b) is the result obtained by the proposed method. The latter has the 'U' shape to help predict the flying plane. The minimal is at −11 in this figure.

(−50, −50, 5), thus the highest point of the ball is higher than the camera. In the case of the real PP position being at (340, 306) and the noise level is 2 pixels, and we use the proposed algorithm to estimate the ball's 3D position. Fig. 9 depicts the result, in which the red curve is ground truth, and the blue curve is the estimated curve.

These experiments verify the proposed algorithm in Section 3. Now, we can conclude that the algorithm is correct in theory.

5.2. Real video data

In the above experiments, the correctness of the algorithm is verified on the synthesized data. Here, we shall test the algorithm on real video. Because these videos for experiments are recorded from broadcast, it is impossible to know the ball's real height and the camera's position; and even we shoot video ourselves, it is too expensive to measure the ball's height. Fortunately, the goalpost height is unique and known in any stadium. Thus, we can test the algorithm through measuring the goalpost height. Fig. 10 shows the estimated goalpost height, in which the red line indicates the real height (2.44 m), and the other curves are the estimated goalpost height on four sequences. For every 10 frames, we calibrate the camera once, and the average camera position is adopted. Only the first frame's homography matrix is computed from the lines' intersections on the playfield, and the other frames' homography matrixes are estimated using (11) based on GME. Point of intersection between any plane passing the stick and the line which passes through the camera position and the virtual shadow of the top of the goalpost is the estimated height. In this experiment, the plane is passing the base line of stadium. From the figure, we can find that the estimated value is close to the real value. For our application, i.e. generating cartoon for highlights segment, it is enough.

5.3. Ball detection and tracking

In this section, we test the algorithm for ball detection and tracking. Fig. 11 shows the procedure of ball detection using Viterbi decode algorithm. In the left column in the figure, the regions surrounded by red rectangles are the detected candidate regions for ball; while the right column shows the ball region, and the appearance of the ball are depicted in the image's left top. The results show that the proposed method determines the ball region in consecutive frames by excluding objects which looks like a ball only in one or some frames.

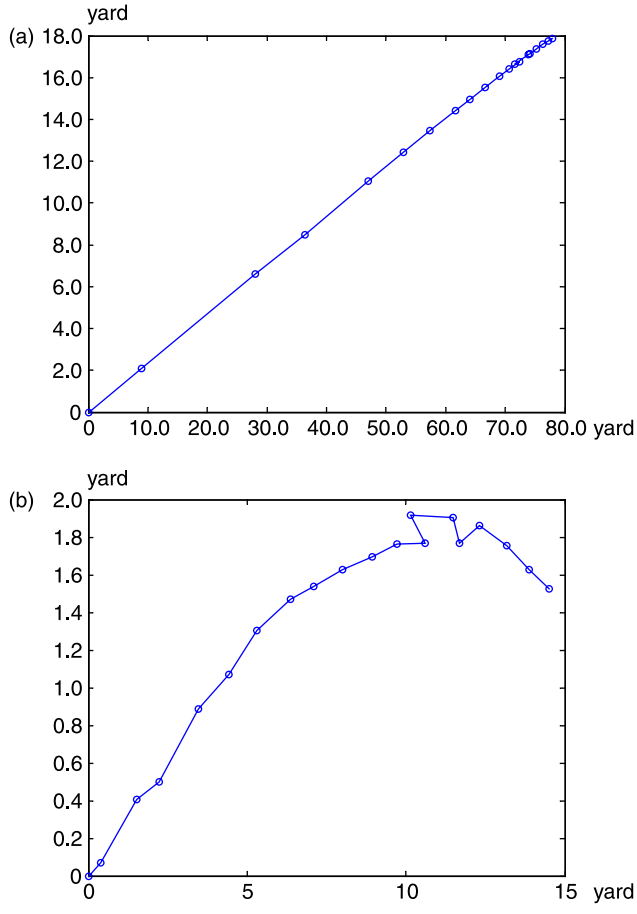

Fig. 13. The estimated balls' height: (a) is the height estimated only through finding the optimal parabola, (b) is the height estimated by the modified method.

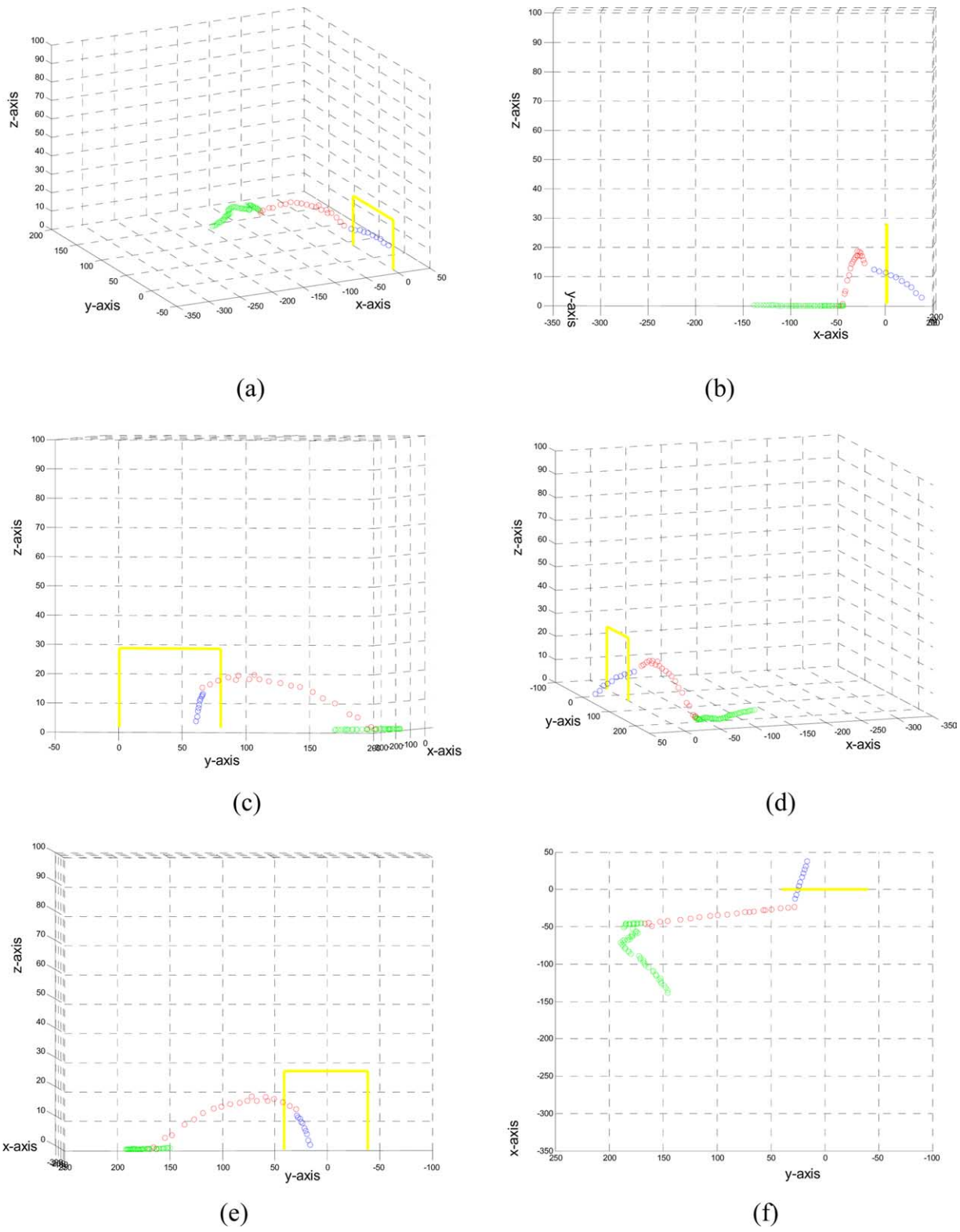

Fig. 14. The ball track from different view point. (a) View from the main camera direction. (b) View along the base line direction. (c) View behind the goal. (d) View from opposite side of the main camera. (e) See from the midfield. (f) View from the top.

The method exploits the assumption that non-ball cannot looks like a ball in multiple frames in general. If this case occurs, it will be difficult even for human to determine ball quickly.

In order to quantify performance of the ball detection and tracking, we label two video sequences with each having over 600 frames. Two criteria, *recall* and *precision*, are adopted to evaluate the algorithm. The larger *recall* is, the more balls in

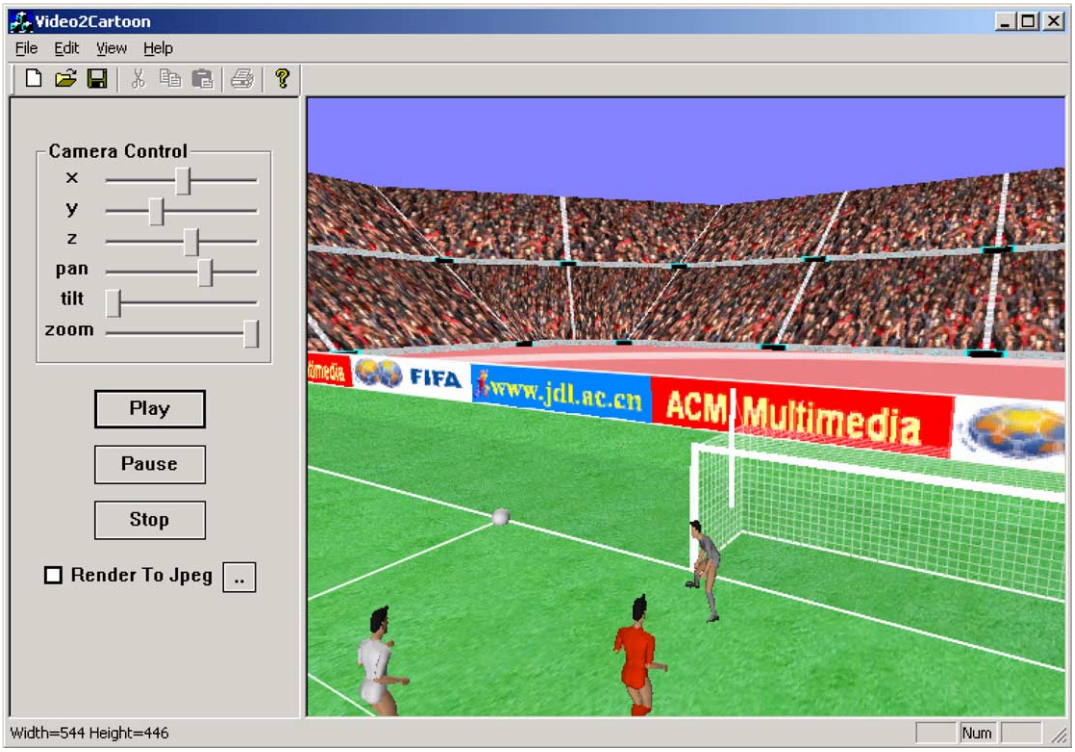

Fig. 15. A snapshot of the prototype of Video2Cartoon.

frames are detected or tracked the higher precision, the more determined ball region comprises the true ball. In the first sequence, the playfield is bad; while it is good in the second one. Table 1 provides the results, respectively. In some parts, such as the second row in Table 1, the recall rate is very low. This is mainly because that the ball is occluded by players, merged with lines, and so on, thus may result the ball detection and tracking to be failure. In some clips, the ball is occluded, and the socks of the player or some line segments are quite similar to the ball, then the precision rate is low. As our assumption is that the ball is nearly circular, the algorithm may fail with the long blurred case. The reason of the performance on good playfield is better than bad playfield is that the segmentation algorithm works better on the former video. Nevertheless, from the table, we can observe that the overall precision and recall rate are acceptable.

As the ball detection method can determine the ball in consecutive frames, we will want to know what if only the detection procedure is used. In the columns of the experiments results, the number before slash is the result of only using the ball detection procedure, and the number behind it is the result of combining the ball detection and tracking procedure. From the table, we can find that the precision of only using detection method is higher a little (less than 5%), since the detection procedure validates the ball needing five consecutive frames. However, the latter method of adding tracking can obtain much higher recall (more than 20%), because the tracking algorithm can continue work even the ball is missing in some frames.

5.4. Estimating moving object position from a highlight segment

We apply the proposed method to extract 3D information from some highlights segments of Europe Cup 2004 (the sequence can be downloaded from http://www.jdl.ac.cn/en/project/mrhomepage/En_demo.htm). Through calibration, the camera is at (−51.8, −56.0, 22.l), and the unit is yard. The players are detected by the method described in this paper and tracked by improved particle filter [5]. In this highlight segment, one player crosses the ball and before it reaches the ground, a player shot it on goal. We can label the ball's jumping-off point; however, the falling point cannot be determined from the video, thus the ball's flying plane has to be predicted by the algorithm proposed in Section 3.4 in the range of (−19, 10, 0) to (29, 10, 0). The optimization function values on different plane are depicted in Fig. 12, in which Fig. 12a is the result by only optimizing the parabola fitting error, and Fig. 12b is the result of the proposed method. We can see that the latter has the shape 'U', so that it can help to predict the flying plane. Fig. 13(a) and (b) shows the ball's height after the crossing and before shot a goal using these two predicted results, respectively. It can be seen that the estimation result of our proposed method is more reasonable. Fig. 14 shows different views of the estimated ball's track in a cube, and the results are estimated by the method in Section 3.4. Since the players are on the playfield, it is easy to compute their real position in real world, so we do not provide the result in detail.

5.5. Application in cartoon generation

At last, we apply the extracted 3D position of ball and players to generate 3D cartoon using OpenGL in VC++, with a snapshot of the system shown in Fig. 15. The generated cartoon can be used to enrich funs or distribute on web. One issue is to estimate the moving objects' positions in real world; the other key problem is to recognize the player's action. This paper exactly focuses on the former problem. For the later issue, at present our system includes two kinds of simple actions in the actions library, walk and run. In the prototype system, users can change the view point, and pan, tilt and zoom the virtual camera. Some sample sequences can be found at http://www.jdl.ac.cn/en/project/mrhomepage/En_demo.htm [33].

6. Conclusion and future work

This paper proposes a method based on self-calibration to estimate the moving objects' 3D positions in broadcast video. The method can estimate not only the players' positions, but also the ball's position even when the ball flies in the air through physical restriction without referring to other known object height. To reduce human interference, the ball and players are detected automatically. Particularly in the ball detection procedure, the method considers consecutive frames, and finds the most likely ball path based on Viterbi decoding algorithm.

Experiments verify the methods, and present promising results. At last, these methods are applied to our project 'Sports video summarization and enrichment (SPISE)'.

In future work, we will apply this method to other field sports, such as volleyball and explore other self-calibration method.

Acknowledgements

The authors will thank to the reviewers' advice. This work is supported by NEC Research China, 'Science 100 Plan of Chinese Academy of Sciences, and Beijing Natural Science Foundation: 4063041'.

Appendix

Let us consider the still point $\mathbf{M}$ in 3D scene, its coordinates are, in camera coordinates system at time $t-1$ and t, $\mathbf{M}_c^{t-1}$ and $\mathbf{M}_c^t$. They have the relationship

$$\mathbf{M}_c^t = \mathbf{R}_{t-1,t}\mathbf{M}_c^{t-1} \tag{A1}$$

where $\mathbf{R}_{t-1,t}$ is a rotation matrix. According to imaging formula, we have

$$x^t = \frac{f_t X^t}{Z^t}, \qquad y^t = \frac{f_t Y^t}{Z^t} \tag{A2}$$

$$x^{t-1} = \frac{f_{t-1} X^{t-1}}{Z^{t-1}}, \qquad y^{t-1} = \frac{f_2 Y^{t-1}}{Z^{t-1}} \tag{A3}$$

where f_{t-1} and f_t are the focal lengths of cameras at time $t-1$ and t. Combining (A1), (A2) and (A3) we can obtain:

$$\begin{aligned} x^t &= \frac{\frac{f_2 r_{11}}{f_1}x^{t-1} + \frac{f_2 r_{12}}{f_1}y^{t-1} + r_{13}}{\frac{r_{31}}{f_1}x^{t-1} + \frac{r_{32}}{f_1}y^{t-1} + r_{33}}, \\ y^t &= \frac{\frac{f_2 r_{21}}{f_1}x^{t-1} + \frac{f_2 r_{22}}{f_1}y^{t-1} + r_{23}}{\frac{r_{31}}{f_1}x^{t-1} + \frac{r_{32}}{f_1}y^{t-1} + r_{33}} \end{aligned} \tag{A4}$$

From (A4), it can be found that the corresponding points of two images are related by the camera's parameter. It is independent of the point position, so we can estimate GME on the whole frame. That is to say, for images captured rotating and zooming camera, they have the relationship of (A5):

$$\tilde{\mathbf{m}}^t \approx \mathbf{P}_{t-1,t}\tilde{\mathbf{m}}^{t-1} \tag{A5}$$

References

[1] J. Assfalg, M. Bertini, C. Colombo, A.D. Bimbo, W. Nunziati, Semantic annotation of soccer videos: automatic highlights identification, Computer Vision and Image Understanding 92 (2–3) (2003) 285–305.
[2] D. Farin, S. Krabbe, P.H.N. de With, W. Effelsberg, Robust camera calibration for sport videos using court models, SPIE Storage and Retrieval Methods and Applications for Multimedia, 2004.
[3] X. YU, X. Yan, T.S. Hay, H.W. Leong, 3D Reconstruction and enrichment of broadcast soccer video, in: Proceedings of the ACM Multimedia, New York, NY, USA, October 2004.
[4] K. Matsui, M. Iwase, M. Agata, T. Tanaka, N. Ohnishi, Soccer image sequence computed by a virtual camera, IEEE Computer Society International Conference on Computer Vision and Pattern Recognition (1998) 860–865.
[5] G.Y. Zhu, D.W. Liang, Y. Liu, Q.M. Huang, W. Gao, "Improving particle filter with support vector regression for efficient visual tracking," International Conference on Image Processing, Genova, Sep. 11–14 (2005) 422–425.
[6] J.C. Ren, J Orwell, G.A. Jones, M. Xu, A general framework for 3D soccer ball estimation and tracking, in: Proceedings of the IEEE International Conference on Image Processing, 2004.
[7] M. Xu, J. Orwell, U. Jones, Tracking football players with multiple cameras, in: Proceedings of the IEEE International Conference on Image Processing, 2004.
[8] S. Iwase, H. Saito, Parallel tracking of all soccer players by integrating detected positions in multiple view images, in: Proceedings of the International Conference on Pattern Recognition, 2004.
[9] H. Saito, N. Inamoto, S. Iwase, Sports scene analysis and visualization from multiple-view video, in: Proceedings of the IEEE International Conference on Multimedia & Expo, 2004.
[10] S. Iwase, H. Saito, Tracking soccer players based on homography among multiple views, in: Proceedings of the Visual Communications and Image Processing, 2003.
[11] H. Kim, S. Hong, Robust image mosaicing of soccer videos using self-calibration and line tracking, Pattern Analysis & Applications 4 (2001) 9–19.
[12] R. Hartley, A. Zisserman, Multiple View Geometry in Computer Vision, second ed., Cambridge University Press.
[13] T. Bebie, H. Bieri, SoccerMan: reconstructing soccer games from video sequences, in: Proceedings of the IEEE International Conference on Image Processing, 1998.
[14] T. Bebie, H. Bieri, A video-based 3D-reconstruction of Soccer games, EuroGraphics, 2000.

[15] S. Choi, Y. Seo, H. Kim, K.S. Hong, Where are the ball and players? soccer game analysis with color based tracking and image mosaick, in: Proceedings of the International Conference on Image Analysis and Processing, 1997.
[16] N. Ancona, G. Cicirelli, E. Stella, A. Ditante, Ball detection in static images with support vector machines for classification, Image and Vision Computing 21 (2003) 675–692.
[17] 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 (2004) 393–408.
[18] 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: Proceedings of the ACM MM 2003, Berkley, CA, USA, November 2–8, 2003, pp. 11–20.
[19] T. Kim, Y. Seo, K.S. Hong, Physics-based 3D position analysis of a soccer ball from monocular image sequence, in: Proceedings of the International Conference on Computer Vision, 1998, pp. 721–726.
[20] I. Reid, A. North, 3D trajectories from a single viewpoint using shadows, in: Proceedings of the British Machine Vision Conference, 1998, pp. 863–872.
[21] Y. Ohno, J. Miura, Y. Shirai, Tracking players and estimation of the 3D position of a ball in soccer games, in: Proceedings of the International Conference on Pattern Recognition, 2000, pp. 145–148.
[22] A. Yamada, Y. Shirai, J. Miura, Tracking players and a ball in video image sequence and estimating camera parameters for 3D interpretation of soccer games, in: Proceedings of the International conference on Recognition, 2002, pp. 303–306.
[23] FIFA, Laws of the game, www.fifa.com/en/regulations/regulation/0,1584,3,00.html
[24] Z. Zhang, Flexible camera calibration by viewing a plane from unknown orientations, in: Proceedings of the Seventh IEEE International Conference on Computer Vision, vol. 1, 20–27 September, 1999, pp. 666–673.
[25] Y. Seo, K.S. Hong, Auto-calibration of a rotating and zooming camera, Proceedings of the IAPR Workshop on Machine Vision Applications, 1998, pp. 274–277.
[26] L. Agapito, E. Hayman, I. Reid, Self-calibration of rotating and zooming cameras, International Journal of Computer Vision 45 (2) (2001) 107–127.
[27] Q. Ye, W. Gao, W. Zeng, Color image segmentation using density-based clustering, in: Proceedings of International Conference on Acoustics, Speech, and Signal Processing, vol. 3, 2003, pp. 345–348.
[28] M. Peura, J. Iivarinen, Efficiency of simple shape descriptors, in: C. Arcelli et al. (Ed.), Advances in Visual Form Analysis, World Scientific, Singapore, 1997, pp. 443–451.
[29] L.R. Rabiner, A tutorial on hidden markov model and selected applications in speech recognition, in: Proceedings of the IEEE, vol. 77(2), 1989.
[30] G. Welch, G. Bishop, An introduction to the kalman filter, TR95-041, UNC at Chapel Hill, 1995, http://www.cs.unc.edu/~welch/kalman/kalmanIntro.html
[31] J. Shi, C. Tomasi, Good features to track, IEEE Conference on Computer Vision and Pattern (1994).
[32] http://www.sourceforge.net/projects/openvlibrary
[33] D.W. Liang, Y. Liu, Q.M. Huang, G.Y. Zhu, S.Q. Jiang, Z.B. Zhang, W. Gao, Video2Cartoon: generating 3D cartoon from broadcast soccer video, Proceedings of the 13th ACM International Conference on Multimedia (ACM MM 2005), Singapore, November 06–11, 2005, pp. 217–218.