Playfield and Ball Detection in Soccer Video

Junqing Yu[1], Yang Tang[2], Zhifang Wang[1], and Lejiang Shi[1]

[1] School of Computer Science & Technology, Huazhong University of Science & Technology, Wuhan, 430074, China
yjqing@hust.edu.cn, terry1014@163.com, shinyaaa@163.com
[2] Department of Development & Planning, Hubei Electric Power Company, Wuhan 430077
ty@hbepc.com.cn

Abstract. The ball is really hard to be detected when it is merged with field lines or players in soccer video. A trajectory based ball detection scheme together with an approach of playfield detection is proposed to solve this problem. Playfield detection plays a fundamental role in semantic analysis of soccer video. An improve Generalized Lloyd Algorithm (GLA) based method is introduced to detect the playfield. Based on the detected playfield, an improved Viterbi algorithm is utilized to detect and track the ball. A group of selected interpolation points are calculated employing least squares method to track the ball in the playfield. An occlusion reasoning procedure is used to further qualify some undetected and false ball positions. The experimental results have verified their effectiveness of the given schema.

Keywords: Playfield detection, Ball detection, Soccer Video.

1 Introduction

As an important specific-domain video genre, sports video has been widely studied for its tremendous commercial potentials. Soccer video is one of the most popular sports ones in the world, so its automatic analysis has become a focus of research efforts. Its possible applications have a broad range, such as indexing, retrieval, annotation, summarization, event detection and tactics analysis. Meanwhile, because the playfield is the place, where almost all the events take place and the ball is the focus for all the audience and players, both playfield and ball detection have been always drawing much attention.

There are mainly two methods to detect the playfield: parametric [1-4] and non-parametric [5, 6]. Gaussian Mixture Model (GMM) is a most widely used parametric method. Estimating parameters and updating model are complicated in GMM. Ordinary Expectation Maximization (EM) algorithm is usually used to estimate the parameters of the GMM. In [1], an unsupervised MAP (Maximum a Posteriori) adaptation is used to adapt GMM to the color of the playfield in each game. Liu [4] proposes IEM (Incremental Expectation Maximization) algorithm to update GMM parameters to adapt the model to the playfield variation with time. Arnaud [5] characterizes the relevant area in the color space and uses a spatial coherence criterion on the selected pixels in the image. In [6], Ekin proposes an algorithm to automatically learn the statistical dominant color of playfield using two color spaces, a control space and a primary space. The information from these two color spaces is combined to be used. As the case with playfield detection, many classical algorithms have been proposed for ball detection and tracking [7-17]. Most of them use Kalman filter to match the ball. And some utilize Viterbi algorithm to detect the bal [13]l, and the Kalman filter based template matching [13] or Condensation algorithm [11] is used to track it.

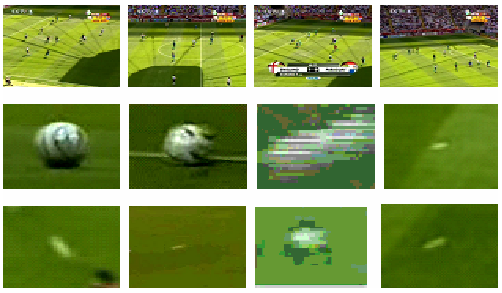


Fig. 1. Typical playfield and ball samples

While, the problem has not yet been fully resolved due to the following challenges:

(1) The playfield may be hard to detect due to the shadow brought out by illumination sunlight;

(2) The ball's features, such as color, shape, and size, are changing with the circumstance conditions like light and velocity. Fig. 1 demonstrates some typical playfield and ball samples in soccer video;

(3) When players possess the ball, it's hardly to segment the ball from the player;

(4) The fragments due to impropriate segmentation of players or field-lines may be similar to the ball;

(5) The ball may be sometimes occluded, and especially when the ball is around the field-lines, it is hard to be segmented because its color is similar to the field-lines.

In this paper, a trajectory based ball detection and tracking scheme with a pre-procedure of playfield detection is proposed in Fig.2. The rest of the paper is organized as follows. In section 2, an improved GLA based playfield detection algorithm is discussed. Improved Viterbi based ball detection algorithm is introduced in section 3. Viterbi algorithm based ball tracking and least square based ball trajectory generating are presented in section 4. In section 5, trajectory based occlusion reasoning is depicted, and some experimental results are presented. Section 6 concludes this paper.

2 Playfield Detection

An improved GLA based method is proposed to detect the playfield color, and Sklansky's algorithm is employed to find the convex boundary of the playfield. Then, the playfield area can be identified through the detected boundary.

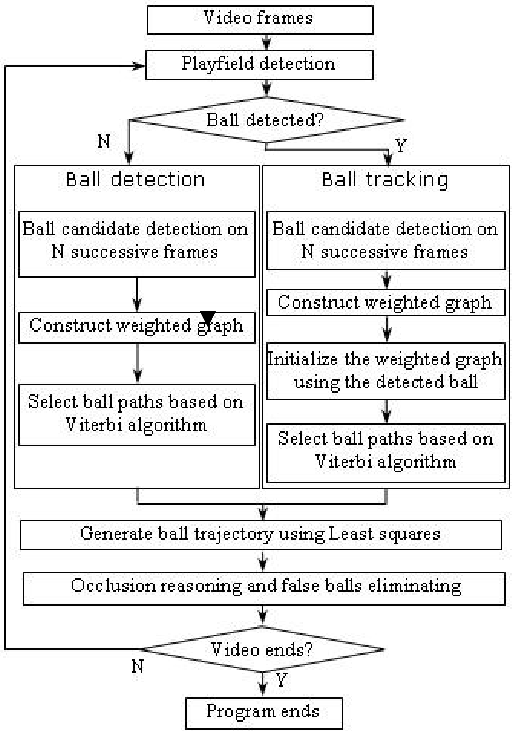

Video frames
Playfield detection
Ball detected?
N
Y
Ball detection
Ball candidate detection on N successive frames
Construct weighted graph
Select ball paths based on Viterbi algorithm
Ball tracking
Ball candidate detection on N successive frames
Construct weighted graph
Initialize the weighted graph using the detected ball
Select ball paths based on Viterbi algorithm
Generate ball trajectory using Least squares
Occlusion reasoning and false balls eliminating
Video ends?
N
Y
Program ends

Fig. 2. Framework of ball detection and tracking

2.1 Playfield Color Identification

Generally, the dominant color of the playfield is green in the soccer video. Therefore, this characteristic can be used to find playfield color. Firstly, the frame color in the video can be classified into different clusters. Then, the cluster containing the majority of pixels can be marked as the playfield color. Every frame of video is a piece of image, which is composed of a set of pixels. The color vectors of the pixels in the image are quantified by the improved GLA. The quantification process is a classification process, so the improved GLA can be employed to classify the pixels. More details of GLA can be referred to [18]. The classification algorithm is designed as follows.

Step 1: Convert the color of an input image from RGB space to HSV space. The playfield dominant color is detected according to the hue and saturation component of the pixel, by which the effect of shadow brought out by illumination can be eliminated. The formulae for converting RGB to HSV color space can be referred to [19].

Step 2: Use clustering method to classify the pixels. At the beginning, each pixel of the input image assigns to one cluster individually.

Step 3: Each pixel is assigned to it's nearest cluster. The distance d between the pixel j and the cluster centroid is calculated by the formula as follows:

$$d = \sqrt{(S_j)^2 + (\overline{S})^2 - 2S_j\overline{S}\cos(\theta(j))} \qquad (1)$$

where, $\theta(j) = \begin{cases} \Delta(j) & if\ \Delta(j) \le 180^0 \\ 360^0 - \Delta(j) & otherwise \end{cases}$, $\Delta(j) = \left|\overline{H} - H_j\right|$

In the above equation, S_j and H_j are the pixel's saturation and hue value; $\overline{S}$ and $\overline{H}$ are the cluster's centroid of saturation and hue value.

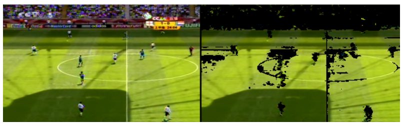
After all the pixels having been classified, cluster centroid should be recalculated to find the mean of the cluster.

Step 4: Calculate the total distortion. If the change in distortion is bigger than the threshold (5%), go to step 3. Otherwise, go to step 5.

Step 5: Calculate the distortion of each cluster. The cluster with the biggest one will be split into two new clusters. If the number of clusters after splitting is more than the upper limited number, go to step 6. Otherwise, go to step 3.

Step 6: Merge the clusters using an agglomerative clustering method. Calculate mutual distances between clusters' centroid to construct a table. If the minimum distance in the table is smaller than the threshold, merge the two relevant clusters and update the table. Otherwise, repeat this step.

Step 7: Finally, the color of the cluster with the most pixels is identified as the playfield color. The detection result of playfield color is demonstrated in Fig. 3. Picture (a) is the original image and (b) is its detected result. The pixels without the playfield color are filled with black.



(a) Original (b) classification result

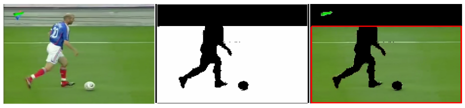
Fig. 3. The detection result of playfield color

2.2 Playfield Boundary Detection

After the playfield color having been identified, the region whose pixel number is less than a given threshold is merged to the neighboring region through region growing process. But the player region affects the playfield detection when it locates on the playfield boundary, as the example in Fig. 4(b). Therefore, a playfield boundary detection algorithm has to be used to solve this problem.

Because the playfield is often a normal polygon, Sklansky's algorithm can be employed to find its convex boundary. Then the playfield area can be filled through the detected boundary. The Sklansky's algorithm has been discussed detailed in [20].

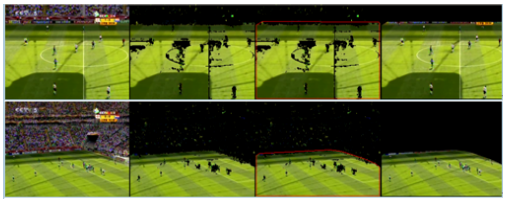
In Fig. 4, (a) is the original image, (b) is the mask of playfield and (c) shows the obtained playfield boundary which has been marked with red line.



(a) Original (b) the mask of playfield (c) playfield boundary

Fig. 4. The detection result of playfield boundary

The whole process of playfield detection is illustrated in Fig. 5. (a) is the original image, (b) is the identified playfield color, (c) is the detected playfield boundary and (d) is the playfield area.



(a) (b) (c) (d)

Fig. 5. Playfield detection results

3 Ball Detection in the Soccer Video

3.1 Ball Candidate Detection

On the green playfield, there are four kinds of objects, including ball, playfield lines, players, and some noises. To find the ball candidates, the key thing is to filter the non-ball objects. To attain this aim, we can use the ball's characteristics, which can differentiate it from other objects. The detailed detection algorithm is explained in the following.

Step 1: Compute the average area ($A_{average}$) of all the objects on the playfield. In order to filter out the players and some tiny noises, only those whose area ranges from $A_{average}/10$ to $2A_{average}/3$ are chosen. Here, 1/10 and 2/3 are set empirically through experiments.

Step 2: Calculate the form factor, which is defined as

$$F = P^2 / (4A) \qquad (2)$$

$$F = P^2 / (4A) \qquad (3)$$

Where P and A refer to the perimeter and of the objects. The bigger the form factor is, the more the object is likely to be a ball. Therefore, the objects with very small form factors can be also filtered out.

Step 3: Compute the ratio of object area to its bounding rectangular area

$$R = A_{\text{area}} / A_{\text{box}} \tag{4}$$

If its ratio is less than 0.2, such object should be filtered out.

Through above three steps, the candidate balls can be detected successfully.

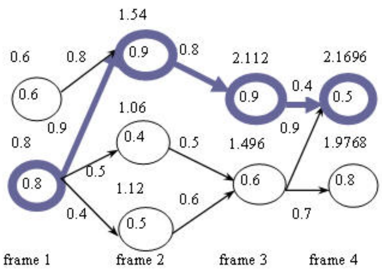
3.2 Graph Construction

A weighted graph is constructed on the ball candidates in the N successive frames. In the graph, nodes denote ball candidates and edges connect those adjacent ball candidates, whose Euclidean distance is smaller than a threshold. Each graph node is assigned a weight to denote its resemblance to the ball, and each edge is assigned a weight to represent the similarity between the connected nodes.

Fig.6 depicts a weighted graph example. Each number inside nodes is node's weight. The number above the node is its cumulative weight denoted by *cweight*, which is the sum of all the node and edge weights on the optimal path ending with the current node. Node weight can be obtained according to the formula (5), while the edge weight can be computed using the similarity between the two connected nodes.

$$weight = 0.5 * F + 0.5 * R \tag{5}$$

Where F means the form factor and R is the ratio A_{area} / A_{box} .



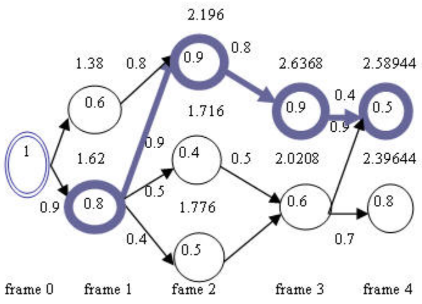


Fig. 6. Weighted graph Illustration **Fig. 7.** Ball tracking algorithm illustration

3.3 Path Selection Based on the Improved Viterbi Algorithm

An improved Veiterbi algorithm is designed to select the ball path, which can be summarized as the following steps. The detailed Viterbi algorithm can be referred in [21].

Step 1: Calculate the cumulative weight (cweight) according to the formula (5), and record the corresponding ex-node, which is the value of superscript *i* in formula (6), for later tracking back use.

Step 2: Get the node with the biggest cweight.

Step 3: Track back to obtain all the nodes on the path according to the ex-nodes recorded in step 1.

Step 4: Check the ends of the selected path. If the path doesn't stretch over all N frames, we can continue select paths respectively among the preceding and succeeding candidates of the current path according to above steps.

$$cweight_j^t = \begin{cases} weight_j^t & (a) \\ \max\limits_{0 \le i \prec N_{t-1}} \{\tau_1 * (cweight_i^{t-1} + \\ weight_j^t) + \tau_2 * weight_{edge}^{ij}\} & (b) \end{cases} \tag{6}$$

In the above formula, $weight_j^t$ means the cumulative weight of the *j*th candidate on frame t. (a) presents the situation that the *j*th candidate has no edge linking to the candidates on its former frame. Otherwise is the (b) situation, where N_{t-1} is the number of candidates on frame *t-1*, $cweight_i^{t-1}$ represents the cumulative weight of the *i*th candidate on frame *t-1*, $weight_j^t$ means the node weight of the *j*th candidate on frame *t-1*, $weight_{edge}^{ij}$ denotes the weight of the edge between the *i*th candidate on frame *t-1* and the *j*th candidate on frame *t*, while τ_1 and τ_2 respectively denote the proportion of the node weight and edge weight in calculating the cumulative weight. In our experiment, τ_1 is set to 0.8 and τ_2 is set to 0.2. The reason why the edge weight has small proportion is that the ball's contour may change greatly over frames because of the change of light and velocity. In Fig. 6, the bold lines and nodes is the optimal path.

4 Ball Tracking and Trajectory Generation

4.1 Ball Tracking

A Viterbi-based algorithm is utilized to track the ball in the successive frames. We construct the weighted graph in the detected ball candidates of N sequential frames. Here, the weighted graph should be initialized using the detected ball. Then, Viterbi-based path selection algorithm can be applied to track the ball. The algorithm is rather similar to that of ball detection. The only difference between them is the initialization procedure. To initialize the weighted graph, the detected ball is added to it as the unique node on the 0th frame, which is assigned a weight of 1. If the distance between the node on the 1st frame and the unique node on the 0th frame is within the threshold, an edge exists. Otherwise, the node can be deleted. The edge weight is assigned according to the similarity of its connected nodes. Fig. 7 demonstrates the ball tracking algorithm. The unique node with weight 1 on the 0th frame represents the detected ball. The bold nodes are the tracked balls.

4.2 Trajectory Generation

Ball Trajectory can be generated based on the raw data points of the detected ball. To generate trajectory function, least squares method is used here. Fig. 8 describes a set of generated functions, where the x-axis denotes time, and the y-axis denotes a dimensional variable, such as the transverse or the vertical displacement of the ball. That is, if the coordinate space is three-dimensioned, then there must be three such sets of trajectory functions. In the Fig. 8, the hollow circles denote the detected ball positions, on which curves are generated using least squares method. The filled circle is the expected ball point computed by the trajectory functions. Therefore, in this process we can not only generation the ball trajectory, but also deleted the false ball and supplement the ignored ones.

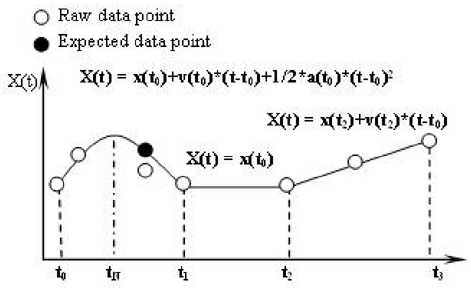

○ Raw data point
● Expected data point
X(t)
X(t) = x(t0)+v(t0)*(t-t0)+1/2*a(t0)*(t-t0)²
X(t) = x(t2)+v(t2)*(t-t0)
X(t) = x(t0)
t0
tN
t1
t2
t3

Fig. 8. Trajectory generation of the detected balls

5 Trajectory-Based Occlusion Reasoning

5.1 Occlusion Reasoning

In the soccer video, the ball is usually be just merged with other objects, moving, possessed by players, or out of the playfield. When the ball is possessed, the player is usually back tracked. When merged only in few frames, the ball position can be directly calculated through the trajectory functions. E.g. in Fig. 8, the ball can not detected in time t_N, but its position can be calculated through $X(t_N) = x(t_0) + v(t_0) * (t_N - t_0) + 1/2 * a(t_0) * (t_N - t_0)^2$, and the corresponding Y(t) and Z(t) can also be calculated in the same way.

5.2 Experimental Results

Experiments have been conducted on 2 soccer video clips with more than 500 frames respectively. The video used is from the matches of 2006 FIFA World Cup. Fig. 9

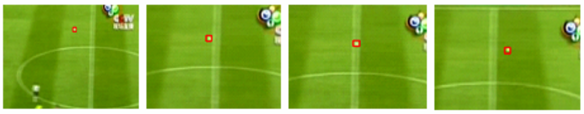


Fig. 9. Ball passing over a field line

shows some frame sequences where the ball passes over a field line. Based on the interpolation function, the ball position has been gained exactly.

In Fig. 10, the player is kicking the ball in sequences a, and heading the ball in sequences b and c.

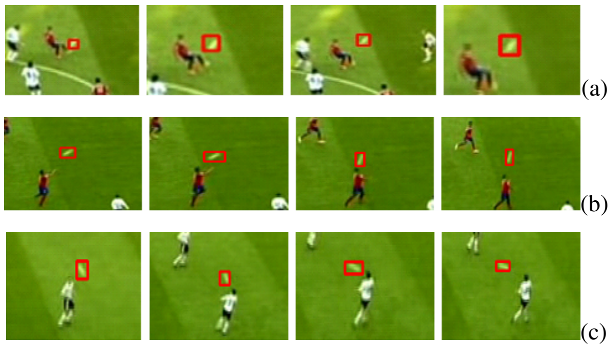


Fig. 10. Ball merged with players

6 Conclusions and Future Work

A trajectory based ball detection and tracking scheme with a pre-procedure of playfield detection has been proposed in the paper. The experimental results have verified that the discussed scheme is reasonable and can be effectively used in soccer video. Using specific-domain approaches, video analysis and semantic extraction become easy. Therefore, our framework is useful to bridge the semantic gap of soccer video. However, our current work does not involve the balls out of the playfield and the event understanding, which are important for automatic analysis of the soccer video. Future work will focus on these topics.

References

1. Barnard, Odobez, J.M.: Robust playfield segmentation using MAP adaptation. In: Proceedings of the 17th International Conference on Pattern Recognition, pp. 610–613 (2004)
2. Jiang, S., Ye, Q., Gao, W., Huang, T.: A new method to segment playfield and its applications in match analysis in sports video. In: Proceedings of the 12th ACM International Conference on Multimedia, pp. 292–295 (2004)
3. Wang, L., Zeng, B., Lin, S., Xu, G., Shum, H.Y.: Automatic extraction of semantic colors in sports video. In: Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing, pp. 617–620 (2004)
4. Liu, Y., Jiang, S., Ye, Q., Gao, W., Huang, Q.: Playfield detection using adaptive GMM and its application. In: Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing, pp. 421–424 (2005)

5. Troter, A.L., Mavromatis, S., Sequeira, J.: Soccer Field Detection in Video Images Using Color and Spatial Coherence. In: Proceedings of International Conference on Image Analysis and Recognition, pp. 265–272 (2004)
6. Ekin, A., Tekalp, A.M.: Robust dominant color region detection and color-based applications for sports video. In: Proceedings of International Conference on Image Processing, pp. 21–24 (2003)
7. Gong, Y., Sin, L.T., Chuan, C.H., Zhang, H., Sakauchi, M.: Automatic Parsing of TV Soccer Programs. In: Proceedings of International Conference on Multimedia Computing and Systems, pp. 167–174 (1995)
8. Seo, Y., Choi, S., Kim, H., Hong, K.: Where are the ball and players? Soccer Game Analysis with Color-based Tracking and Image Mosaic. In: Proceedings of 9th International Conference on Image Analysis and Processing, (2), pp. 196–203 (1997)
9. Ohno, Y., Miura, J., Shirai, Y.: Tracking Players and Estimation of the 3D Position of a Ball in Soccer Games. In: Proceedings of 15th International Conference on Pattern Recognition, (1), pp. 145–148 (2000)
10. D'Orazio, T., Ancona, N., Cicirelli, G., Nitti, M.: A Ball Detection Algorithm for Real Soccer Image Sequences. In: Proceedings of 16th International Conference on Pattern Recognition, pp. 201–213 (2002)
11. Yow, D., Yeo, B.L., Yeung, M., Liu, B.: Analysis and Presentation of Soccer Highlights from Digital Video. In: Proceedings of Asian Conference on Computer Vision, pp. 499–503 (1995)
12. Tong, X., Lu, H., Liu, Q.: An Effective and Fast Soccer Ball Detection and Tracking Method. In: Proceedings of the 17th International Conference on Pattern Recognition, (4), pp. 795-798 (2004)
13. Liang, D., Liu, Y., Wang, Q., Gao, W.: A Scheme for Ball Detection and Tracking in Broadcast Soccer Video. In: Proceedings of Pacific-Rim Conference on Multimedia, (1), pp. 864–875 (2005)
14. Yu, X., Xu, C., Tian, Q., Leong, H.W.: A Ball Tracking Framework for Broadcast Soccer Video. In: Proceedings of International Conference of Multimedia and Expo, (2), pp. 273–276 (2003)
15. Yu, X., Tian, Q., Wan, K.W.: A Novel Ball Detection Framework for Broadcast Soccer Video. In: Proceedings of International Conference of Multimedia and Expo, (2), pp. 265–268 (2003)
16. Yu, X., Xu, C., Leong, H.W., Tian, Q., Tang, Q., Wan, K.W.: Trajectory-Based Ball Detection and Tracking with Applications to Semantic Analysis of Broadcast Soccer Video. In: Proceedings of the 11th ACM International Conference on Multimedia, pp. 11–20 (2003)
17. Ren, J., Orwell, J., Jones, G.A.: Generating Ball Trajectory in Soccer Video Sequences. In: Workshop on Computer Vision Based Analysis in Sport Environments, Graz, Austria (2006)
18. Gersho, A., Gray, R.M.: Vector quantization and signal compression [M]. Kluwer Academic Publishers, Boston (1992)
19. Gonzalez, R.C., Wintz, P.: Digital Image Processing [M], 2nd edn. Addison-Wesley, Reading, MA (1987)
20. Sklansky, J.: Finding the convex hull of a simple polygon [J]. Pattern Recognition Letters 1(2), 79–83 (1982)
21. Rabiner, L.R.: A Tutorial on Hidden Markov Model and Selected Applications in Speech Recognition. Proceedings of the IEEE 77(2), 257–286 (1989)