Proc. ECCV-2006 Workshop on Computer Vision Based Analysis of Sports Environments (CVBASE'06), pp. 26-37, 2006.

Ball Route Estimation in Broadcast Soccer Video

Takumi Shimawaki[1], Jun Miura[1], Takuro Sakiyama[1], and Yoshiaki Shirai[2]

[1] Department of Mechanical Engineering, Osaka University,
Suita, Osaka 565-0871, Japan
{simawaki, jun, saki}@cv.mech.eng.osaka-u.ac.jp
http://www-cv.mech.eng.osaka-u.ac.jp/

[2] Department of Human and Computer Intelligence, Ristumeikan University,
Kusatsu, Shiga 525-8577, Japan
shirai@ci.ritsumei.ac.jp

Abstract. This paper deals with the analysis of broadcast soccer video. To recognize interesting events such as a goal, estimation of ball movements is necessary. It is, however, sometimes difficult to detect a ball by a simple color and shape-based method when it overlaps with players and lines. We therefore develop a method of estimating a ball route during such overlaps by considering spatio-temporal relationships between players, lines, and the ball. The method can deal with difficult cases such as the one where a ball disappears at a player and re-appears from another player. Experimental results show the effectiveness of the method.

1 Introduction

There are increasing demands for summarizing a broadcast video of a soccer game (or other sports) to make a digest of interesting scenes such as goal scenes so that viewers can quickly survey the game. We are now developing a system for retrieving interesting and informative scenes based on scene understanding. To understand various scenes in soccer games, it is essential to know the movements of players and a ball. This paper focuses on ball tracking.

There are several ball detection methods, which use, for example, SVM [1] or a generalized Hough transform [2]. Most of the previous methods are, however, applicable only when a ball is sufficiently large and not so fast in the image. In our case, a ball is usually small and sometimes moves fast. Moreover, it sometimes overlaps with players and lines.

Several ball tracking methods deal with such problems by applying statistical filters such as Kalman filter [3] and particle filter [4] or by using ball trajectory models [5, 6]; these methods can handle short-term occlusions or overlaps. In actual scenes, however, players and lines often overlap with each other and, as a result, a ball which has been overlapped with (or occluded by) a player may appear from another player's region. To cope with such cases, it is necessary to examine possible *routes of the ball* (i.e., a sequence of objects which overlap with (or occlude) the ball) based on spatio-temporal relationships between players, lines, and the ball.

This paper proposes a method of estimating the route of a ball when it overlaps with players and lines in broadcast soccer video. We focus on the images taken from the center camera which spans the widest area of the field. We automatically detect shots from that camera and estimate the camera parameters during the shots. Using the estimated parameters, all image data are transformed into a fixed image coordinate system, where all analyses are performed.

The rest of the paper is organized as follows. Sec. 2 describes the parameter estimation. Sec. 3 briefly explains how to track players, and how to track a ball while it does not overlap with players and lines for a long period. Sec. 4 describes the ball route estimation during overlaps in detail. Sec. 5 summarizes the paper and discusses future works.

2 Camera Parameter Estimation

The position of the camera is estimated once in advance by manually matching the lines and the frame of the goal in the image with those of a model of the field in the scene. The other three parameters (pan, tilt, and zoom) are estimated online since they usually change frame to frame. This section explains an on-line parameter estimation, which is composed of the following three parts:

– Detection of a shot from the center camera.
– Estimation of initial camera parameters at the first frame of the shot.
– Estimation of camera parameters in subsequent frames. This part uses a previously-developed estimation method based on a local search in the parameter space [7].

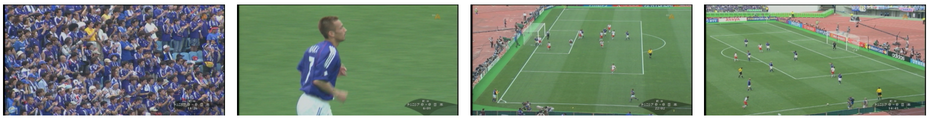
2.1 Detecting Shots from the Center Camera

We use a color histogram of 48 bins (16 bins for each color (R, G, B)) for shot change detection. If the sum of the differences between the corresponding bins in the current and the previous frame exceeds a threshold, the current frame is considered to be the first frame of a new shot. This first frame is analyzed to determine if the shot is taken by the center camera.

We use the following three conditions for this determination:

1. A large part of the image is the ground. If regions of the ground color occupy more than 40% of the image, this condition is satisfied.
2. Sizes of players are sufficiently small. If the size of the largest region which is in the ground region and does not have the color of lines (i.e., white) is less than 1% of the image, this condition is satisfied. This condition is for removing shots of zoomed-in players.
3. The number of horizontal lines is less than or equal to two and that of vertical lines is less than or equal to one. This condition is for removing shots from the corner cameras (used for capturing corner kick scenes).

For four images in Fig. 1, for example, the first, the second, and the third condition remove (a), (b), and (c), respectively. Note that in Fig. 1(c), there are two vertical and three horizontal lines, while in Fig. 1(d), no such lines exist.



(a) the stands (b) zoomed-in player (c) from left camera (d) from center camera

Fig. 1. Shot examples.

2.2 Estimating Camera Parameters in a Shot

The initial values of the three camera parameters can be estimated from two pairs of intersections of lines in the image and those of the field model. The steps for the estimation is as follows.

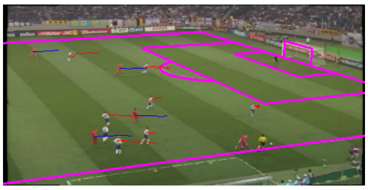
1. Choose three lines in the image among the ones extracted for shot detection such that at least one line is not in parallel with the others, and determine the position of two intersections.
2. For each line in the image, determine a set of possible corresponding lines in the model by considering the possible ranges of the gradients of the model lines *in the image*, determined from the movable ranges of the camera parameters.
3. Generate hypotheses of line-to-line correspondence by considering all combinations of correspondence of the three lines. Each hypothesis is composed of three pairs of line-to-line correspondence.
4. For each hypothesis:
   (a) Calculate the positions of two intersections in the model.
   (b) Calculate the camera parameter from two pairs of intersections in the image and in the model.
   (c) Project all model lines into the image using the calculated parameter, and see how these projected lines match with the line regions (i.e., white regions) in the image.
5. Select the best parameter which maximizes the degree of the matching between model lines and line regions.

Once the initial parameters are determined, the parameters are continuously estimated in subsequent frames using a local search-based method [7]. Fig. 2 shows a result of the camera parameter estimation. Fig. 2(a) indicates the result of projecting model lines onto the ground using the estimated parameters. In Fig. 2(b), the bright area corresponds to the field of view of the camera, and red and blue points indicate the positions of players.

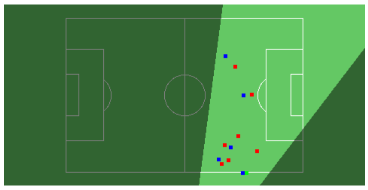
3 Player and Ball Tracking

3.1 Detecting and Tracking Players

The colors of uniform in the HSV space are registered in advance. We use this color information to extract shirts and pants regions whose sizes are within a



(a) projected lines.



(b) Field of view and player positions.

Fig. 2. A result of camera parameter estimation.

certain range. The range is changed according to the estimated focal length and the player's position on the field. We search for pairs of the shirts and the pants region which align almost vertically in the image.

In tracking a player, we predict the position of the player on the field by a simple linear extrapolation from the previous two positions. We project the predicted position onto the image by using the estimated camera parameters and search a neighboring area of the projected position for the player.

If two players overlap, we detect the overlap and track both players as one region until the players are separated. We determine which one is occluding by color in the case of players of different teams and by the vertical position in the image (i.e., the lower player occludes the upper one) in the case of players of the same team. A snapshot of player tracking is shown in Fig. 2.

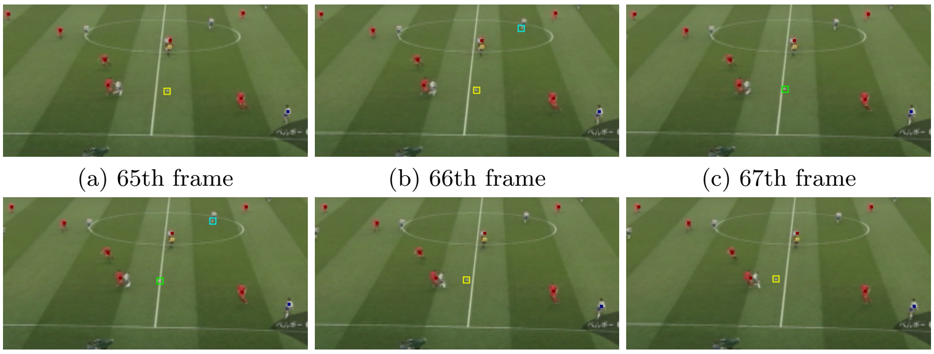
3.2 Detecting and Tracking a Ball

We use images of 480×240 pixels in which the ball is sometimes too small to reliably detect from a single image. We therefore extract ball candidate regions every frame and see if they form a continuous movement. A ball candidate region is a white region inside the ground whose size and aspect ratio are within some predetermined ranges. For each ball candidate region in a frame, we search its neighbor in the next frame for candidate regions. If we find candidate regions in three consecutive frames, we consider a ball is detected.

Once a ball is detected, we track it using a simple prediction as follows.

1. Predicted position $\boldsymbol{x}(t)$ at frame t is given by the following:
$$\boldsymbol{x}(t) = \boldsymbol{x}(t-1) + \{(\boldsymbol{x}(t-1) - \boldsymbol{x}(t-2)) + (\boldsymbol{x}(t-2) - \boldsymbol{x}(t-3))\}/2$$
2. If a ball candidate is extracted within a neighbor of the predicted position, it is considered to be the ball.
3. If no candidates are found, we keep tracking with calculating the predicted positions repeatedly. If we do not find ball candidate for five consecutive frames, we consider that the ball disappeared (occluded or overlapped).

Fig. 3 shows a successful result of ball tracking with a short overlap. The yellow boxes indicate tracked ball, the green ones indicate the predicted positions when a ball candidate is not extracted, and the blue ones indicate other ball candidates.



(a) 65th frame (b) 66th frame (c) 67th frame

(d) 68th frame (e) 69th frame (f) 70th frame

Fig. 3. A result of ball tracking.

(a) player (b) line (c) the stands

Fig. 4. Example cases where a simple ball detection fails.

4 Ball Route Estimation during Overlaps

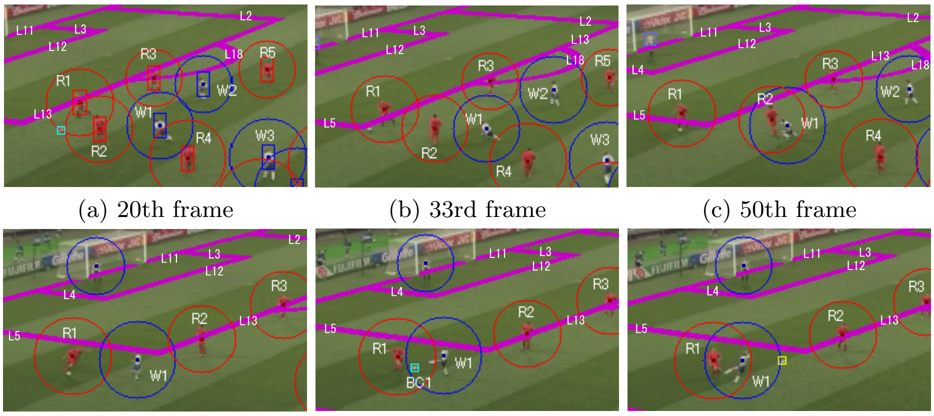
We consider the following three cases where a ball cannot be detected using a simple ball detector (see Fig. 4).

1. Overlap with players (or referees) (see Fig. 4(a)).
2. Overlap with lines (see Fig. 4(b)).
3. Overlap with the stands (see Fig. 4(c)).

As mentioned above, a difficult situation is that a ball continuously overlaps with several players and lines. Fig. 5 shows such a situation: (a) a ball is detected near two red players (R1 and R2); (b) the ball overlaps with a line (L13) and tracking is suspended; (c)-(d) a red player (R1) keeps the ball; (e) he kicks the ball; (f) the ball is detected again at the back of the white player (W1).

We examine the frames between the disappearance and the re-appearance of a ball to estimate the ball route in the following steps.

1. Enumerate possible transitions of the ball between objects (players, lines, or the stands) that overlap with the ball.
2. Generate ball route candidates considering spatio-temporal relationships between the objects and the ball.
3. Generate a rough ball trajectory for each ball route candidate, if possible, by considering constraints on ball movements.



(a) 20th frame (b) 33rd frame (c) 50th frame

(d) 80th frame (e) 87th frame (f) 93rd frame

Fig. 5. An example scene where a ball has not been detected for a long period.

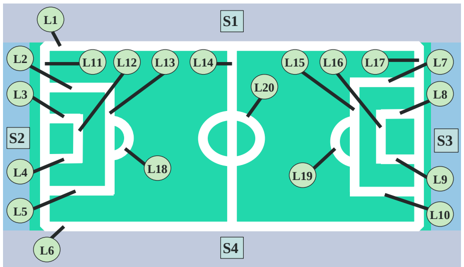


Fig. 6. Division of lines and the stands.

4. Evaluate the trajectories based on the detection of ball-like regions on them, and select the best trajectory and thus the best ball route.

4.1 Enumerate Ball Transitions

We construct a graph called a *transition graph* that enumerates possible transitions of a ball between objects. Nodes of the graph consist of objects that may overlap with the ball, that is, players, lines, and the stands. A *ball candidate*, which is an isolated ball candidate, is also represented as a node. Links consist of possible transitions between the nodes. Lines on the ground are divided into straight lines and curved ones; the stands are divided into four regions. Fig. 6 shows the nodes of lines and the stands. We consider the following transitions:

1. player $\longleftrightarrow$ player, ball candidate, line.
2. ball candidate $\longleftrightarrow$ line.
3. line or the stands $\longleftrightarrow$ line or the stands

The transitions including players and ball candidates are *temporal* and effective only while two nodes are close enough. In Fig. 5, the circle drawn around

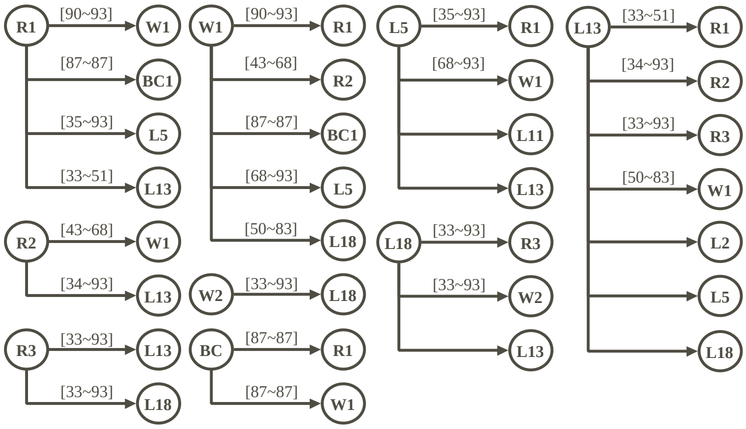


Fig. 7. Part of transitions between nodes from the sequence shown in Fig. 5.

each player shows the range within which the ball can move in the next frame. If there is another player's centroid or a line in this circle, the ball may make transition to one of them. Transition between lines and the stands are *fixed*; transitions between adjacent nodes in Fig. 6 are possible.

Fig. 7 shows a part of transitions generated from Fig. 5. Labels of nodes, W*, R*, L*, BC* indicate white players, red players, lines, and ball candidates, respectively. For example, since a white player (W1) exists near a red player (R1) during frames 90–93 (see Fig. 5(f)), transition R1→W1 for that period is generated; also, since R1 is near a line (L5) during frames 35–93 (see Fig. 5(c)-(e)), transition R1→L5 is generated.

4.2 Generate Ball Route Candidates

We generate ball route candidates by searching the transition graph for possible routes connecting the node where a ball disappears and the node where the ball re-appears. In this candidate generation, we consider the temporal consistency of transitions. That is, the earliest frame of the transition that gets into a node should be earlier than the latest time of the transition that gets out of the node. We also use the following rules to avoid generation of unrealistic transitions.

1. One player node can appear only once in a route. This is to avoid ball movements in which a ball moves back and forth between the same players.
2. The maximum number of successive line nodes is two. In addition, the shape of the two successive lines should match to a physically-possible ball movement. In Fig. 5(b), for example, the transition R1→L13→L18→W2 is possible because the lines L13 and L18 can be approximated by a parabola in the image and the ball movement can sometimes be parabolic.
3. The maximum number of successive stands nodes is two.

Fig. 8 shows some of generated ball route candidates from Fig. 7.

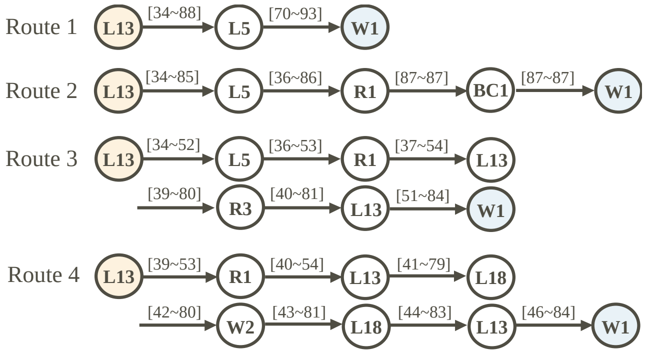


Fig. 8. Some of generated ball route candidates from Fig. 7.

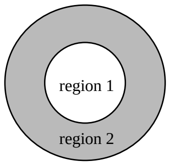


Fig. 9. Separability filter.

4.3 Generate and Evaluate Rough Ball Trajectories

Detection of ball candidates during an overlapping period is difficult. Nevertheless we would like to have evidence to be used to rank ball route candidates. We therefore search for ball-like regions using *separability filter* [8] inside the regions suggested by each ball route candidate. Assuming that a ball *approximately* exhibits a linear motion in the image for a relatively short period of time, we extract a set of ball-like regions that forms a linear segment. We then combine such segments to construct a rough ball trajectory that matches with one of the ball route candidates. Each route is evaluated based on the reliability of the corresponding rough ball trajectory.

Extraction of Ball-Like Regions We search the region determined by each ball route candidate for ball-like regions with separability filter. Separability filter responses to cocentric circular patterns like Fig. 9 and is often used for detecting eyes in face recognition. It outputs separability value η defined as:

$$\eta = \begin{cases} \eta' & (\overline{I_1} \geq \overline{I_2}) \\ -\eta' & (\overline{I_1} < \overline{I_2}) \end{cases} \qquad \eta' = \frac{n_1(\overline{I_1} - \overline{I_m}) + n_2(\overline{I_2} - \overline{I_m})}{\sum_{i=1}^{N}(I_i - \overline{I_m})^2},$$

where n_1 and n_2 the numbers of pixels of region 1 and region 2, respectively; $N = n_1 + n_2$; I_i is the brightness of pixel i; $\overline{I_1}$, $\overline{I_2}$, and $\overline{I_m}$ are the averaged brightness of region 1, region 2, and the whole region, respectively.

The regions that have higher responses to the separability filter than some threshold are extracted. Since white players' shirts and socks regions output high responses, we remove shirts regions using the result of player tracking. We also remove socks regions by examining their shape; if the ratio of the longer principal axis to the shorter one of a region is larger than some threshold, the region is considered to be a socks region. Fig. 10(a) shows the filter output after the removal of shirts and pants regions. Fig.10(b) shows the ball-like regions inside a part of the search region. Since several white regions other than the ball


(a) filter output.

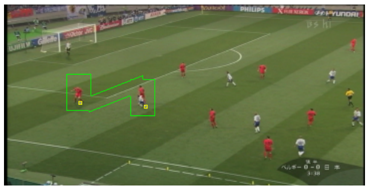

(b) extraction result.

Fig. 10. Extraction of ball-like regions.

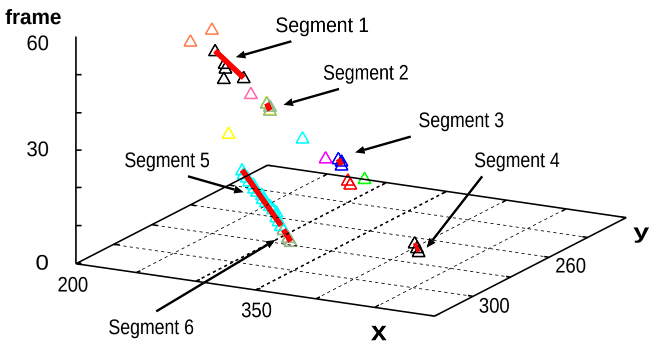


Fig. 11. Clustering of ball-like regions and segment generation.

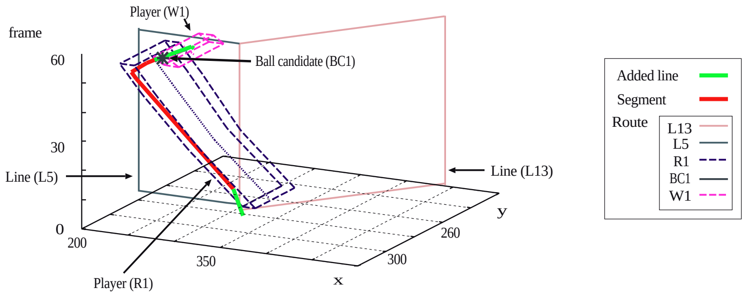
region may remain as shown in Fig. 10(b), we use the motion continuity to filter out such regions, as described below.

Generation of Sequences of Ball-Like Regions We generate sequences of ball-like regions (called *segments*). We first perform a simple clustering of the regions; if two regions are within a certain distance *in space and time*, they are put in a cluster. Clusters with less than three regions are deleted. We then fit a line to each cluster to generate a segment. Fig. 11 shows a result of clustering and segment generation for Route 3 in Fig. 8. Triangles are extracted ball-like regions and their colors indicate cluster IDs; lines indicate generated segments.

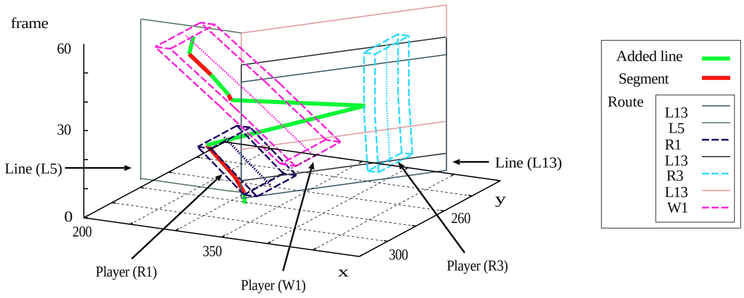
The sum of the outputs of the separability filter for the regions in a segment is called the *score* of the segment and is used for selecting the most probable ball route, as described below.

Generation of Rough Ball Trajectories We generate a set of rough ball trajectories during the overlapping period; a trajectory should pass all nodes of the ball route candidate under consideration and include at least one segment.

In rough trajectory generation, we first enumerate all possible combinations of segments such that no more than one segment exist at a time. We then try to generate a trajectory for each combination. A trajectory is composed of segments



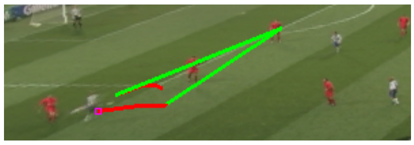
(a) Trajectory for Route 2 in Fig. 8.



(b) Trajectory for Route 3 in Fig. 8.

Fig. 12. Examples of generated trajectories.

(a) Ball movement for Fig. 12(a).



(b) Ball movement for Fig. 12(b).

Fig. 13. Ball movements projected onto the mosaicked image.

and straight lines added to connect the segments. The lines should satisfy the constraint of the maximum ball speed.

Fig. 12 shows two rough ball trajectories generated from the segments shown in Fig. 11, for Routes 2 and 3 in Fig. 8. Each trajectory is generated so that it passes the nodes (players, lines, and ball candidates) in the corresponding routes. Red bold lines and green ones indicate the segments (i.e., sequences of extracted ball-like regions) and the added lines, respectively. The existence of players and lines in space-time is also shown. Fig. 13 illustrates how the ball moves in the mosaicked image in the case of Route 2 and 3.

Selection of Most Probable Ball Route The score of a feasible trajectory is the sum of the scores of its segments normalized by the total length of

Table 1. Scores of routes in Fig. 8.

ball route candidate	score	sum of filter outputs	trajectory length
Route 1	0.00	0	75
Route 2	22.24	3403	153
Route 3	4.00	1204	301
Route 4	4.12	1832	445

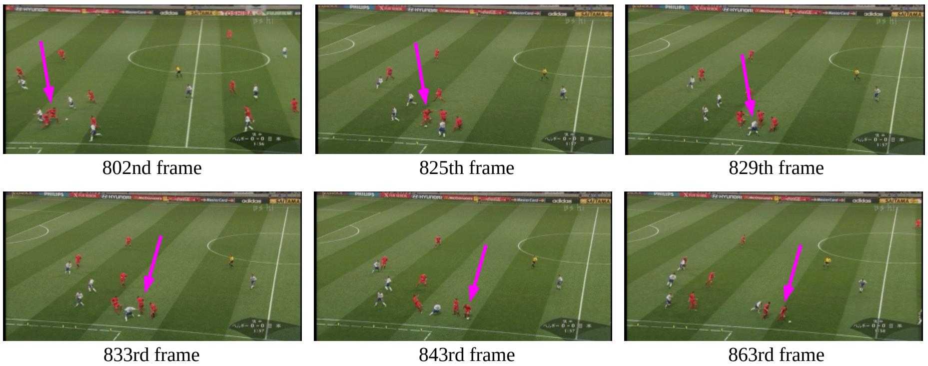


Fig. 14. Another result of ball route estimation.

the trajectory. For each ball route candidate, we select the trajectory among its possible feasible trajectories which has the highest score; this score is the score of the ball route candidate. We finally select the ball route with the highest score. Table 1 summarizes the scores of the four routes in Fig. 8; Route 2 (L13→L5→R1→BC1→W1) is finally selected.

4.4 Results for Other Sequences

Fig. 14 shows another sequence for which the proposed method can estimate the ball route correctly. Pink arrows indicate the nodes which are estimated to have the ball at each frame. The estimated route include three red players and one white player.

We also tested the method for a long soccer video of about nine and a half minutes. The total number of frames in the video is 17,279. The method automatically detected five shots taken by the center camera; the number of frames for the shots is 2,289 in total. In the shots, there are 34 sequences during which a ball is not detected by a simple method and to which the proposed method is thus applied. Their average number of frames is about 20. We examined the outputs of the method for these sequences and found the estimated route is correct for 28 sequences. Failures in ball route estimation are mostly caused by those in detecting ball-like regions, especially when a ball goes out of the field into the stands immediately after a player kicks or heads it. Detection of ball-like regions needs to be improved to cope with such cases.

5 Conclusions and Discussion

This paper has presented a method of estimating a ball route in soccer broadcast video when a ball continuously overlaps with players and lines. We first generate a transition graph representing possible transitions of the ball between overlapping objects, based on their spatio-temporal relationships. We then enumerate ball route candidates from the graph and select the best one by searching for the evidence for ball existence near each route candidate. By this two-stage approach, we can greatly reduce the region to examine in the image. We have also described a method of on-line estimating the camera parameters. The method exhibits a good performance for a set of difficult scenes.

The current method estimates a ball route *in the image*; when a ball passes a player, for example, the method does not tell whether or not the ball is actually touched by the player. To make such a judgment, we need additional inference about the actual trajectory of the ball and the positions of the players on the ground. This is a future work. Another future work is to apply the method, with necessary improvements, to the shots other than the ones from the center camera. This is necessary for developing a scene retrieval and summarization system.

References

1. N. Ancona, G. Cicirelli, E. Stella, and A. Distante. Ball Detection in Static Images with Support Vector Machines for Classification. *Image and Vision Computing*, Vol. 21, pp. 675–692, 2003.
2. T. D'Orazio, C. Guaragnella, M. Leo, and A. distante. A New Algorithm for Ball Recognition Using Circle Hough Transform and Neural Classifier. *Pattern Recognition*, Vol. 37, pp. 393–408, 2004.
3. X. Yu, C. Xu, Q. Tian, and H. W. Leong. A Ball Tracking Framework for Broadcast Soccer Video. *Proc. of Int. Conf. on Multimedia and Expo*, Vol. 2, pp. 273–276, 2003.
4. K. Choi and Y. Seo. Probabilistic Tracking of the Soccer Ball. *Proc. of ECCV Workshop on Statistical Methods in Multi-Image and Video Processing*, pp. 50–60, 2004.
5. Y. Ohno, J. Miura, and Y. Shirai. Tracking Players and Estimation of the 3D Position of a Ball in Soccer Games. *Proc. 15th Int. Conf. on Pattern Recognition*, pp. 145-148, 2000.
6. J. Ren, J. Orwell, G.A. Jones, M. Xu. A General Framework for 3D Soccer Ball Estimation and Tracking. *Proc. IEEE Int. Conf. on Image Processing*, 2004.
7. A. Yamada, Y. Shirai, and J. Miura. Tracking players and a ball in video image sequence and estimating camera parameters for 3d interpretation of soccer games. *ICPR*, pages 303–306, 2002.
8. K. Fukui and O. Yamaguchi. Facial Feature Point Extraction Method Based on Combination of Shape Extraction and Pattern Matching. *Systems and Computers in Japan*, Vol. 29, No. 6, pp. 696–710, 1999.