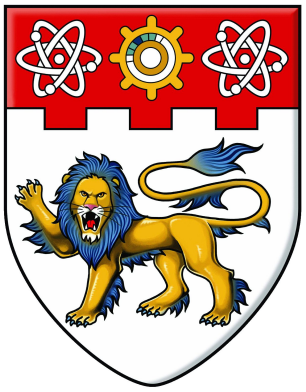

NANYANG TECHNOLOGICAL UNIVERSITY

Real Time Automated Analysis of Soccer Videos

Submitted By: Muhammad Haziq Bin Razali
Matriculation Number: U1322810E

Supervisor: Dr. Stefan Winkler
Co-Supervisor: Dr. Wang Gang

A final year project report submitted to Nanyang Technological University
In partial fulfilment of the requirements of the degree of
Bachelor of Engineering

2016

Abstract

In a soccer match, the ball is invariably the focus of attention. While there have been many works dedicated to object tracking, the tracking of soccer ball remains a challenge due to the lack of features, its relatively small size and the occlusions that occur during player-ball interaction. In this thesis, we present an automated real time tracking system for soccer videos utilizing multiple cameras for complete coverage of the soccer field. A background subtraction algorithm serves as the first stage in the object detection framework from which the detected objects are classified as either a ball or a player through a combination of multiple techniques. Player labelled objects are subsequently tracked via a Kalman filter with an adaptive template for occlusion handling. The ball tracking algorithm is based on a template matching algorithm and takes into account prior knowledge of soccer for occlusion and event handling. Results from each camera are lastly registered onto a model of the soccer field for analysis in 3D. Experiments on the ISSIA soccer dataset show that the system is effective with promising precision and recall measures.

Acknowledgments

I would like to express my sincere gratitude to both Dr. Stefan Winkler and Professor Wang Gang for their continuous support during this project. Their patience, motivation, and immeasurable knowledge have helped steer me in the right direction. I could not have imagined having better advisors and mentors during this last leg of my undergraduate journey.

Contents

Chapter 1

Introduction

Video processing has found many applications in sports such as soccer. Being an incredibly competitive field, clubs around the globe are incorporating video analysis methods as training tools in the development of the team. In the context of team development, playbacks provide unparalleled coverage of key events, enabling teams to understand their own strengths and weaknesses, facilitating strategy development. Having said that, there exists an unmet need to automate and improve the analysis of soccer videos as all related works require some sort of manual input to annotate important events and to perform statistical analysis.

In this thesis, we present an automated real time tracking system for soccer videos. The system utilizes multiple static cameras for complete coverage of the soccer field. The objectives are to develop robust methods for the tracking of players and the ball across multiple cameras. In addition, since the ball is often flying across the air, robust and effcient methods to localize the ball in 3D must be investigated.

The organization of the remainder of this thesis is as follows. We start off by presenting a review of related works on multi and single camera ball tracking in chapter 2. Their approach will be summarized and their strengths and weaknesses discussed. In chapter 3, the outline of the system will be presented diagrammatically. Chapter 4 begins describing the system by covering the object detection framework. This contains a comprehensive description of the algorithms used to locate all objects of interest. Following that, the object tracking architecture will be delineated in chapter 5 where we explain the algorithms used. Given all the tracked objects from each camera, we must then localize the positions of the players and the ball with respect to world coordinates and ensure that only valid

correspondences are generated. This will be covered in chapter 6. Finally chapter 7 will present the performance of the designed tracker before we conclude in chapter 8.

Chapter 2

Related Works

In the soccer domain, various methods for the detection, tracking and localization of the soccer ball in 3D have been proposed. Ohno et als [1] multi camera setup incorporated frame differencing and trajectory mining to identify ball candidates in each separate view. Objects from consecutive frames are labelled as candidates if found within close proximity. This process is repeated until one candidate is selected as a ball.

Similarly, J. Ren et al [2] modelled a background image via the Gaussian Mixture Model before identifying candidates based on their size and velocity over multiple frames. The 2D ball positions from different camera views are then integrated to obtain a 3D position using motion models [1] or geometry [2].

The downside of such approaches for the detection and tracking of a soccer ball is that the number of frames required until a candidate is selected is dependent on the severity of noise. Additionally, trajectory mining or motion analysis is impossible in the presence of heavy occlusion especially during player ball interaction as the output of frame differencing in [1] and background subtraction in [2] would give objects that are merged.

Kim et al [3] and Reid et al [4] presented different techniques to estimate the position of the ball in 3D with a single camera by utilizing reference players and shadows. These techniques are unlikely robust however as the shadow positions are influenced by light source rather than on camera projections.

Matsumoto et al [5] utilized 4 cameras employing background subtraction and template matching framework to identify ball candidates in each view although

their research focused on finding an optimized viewpoint given the tracks and did not include any analysis in 3D.

For the detection and tracking of soccer ball without any analysis in 3D, the techniques presented in [5, 6, 7, 8, 9, 10, 11] all follow a similar approach in which multiple candidates are generated over consecutive frames before declaring one the real ball. To conclude, while there have been many related works on ball detection and tracking in the context of soccer videos, they all require the object to be detected over multiple frames before it is declared as a ball. Additionally, none of them have used multiple cameras as a means to enhance the performance of the detector.

Chapter 3

Outline of Approach

The ISSIA soccer dataset [12] is used to develop the system. It consists of 6 camera views (Figure 3.1) of the playfield at 25 frames per second with a resolution of 1920 x 1080 pixels. It is also accompanied by 6 xml files; each containing the ground truth of the player, referee and ball positions in pixels for each camera view.

The framework for the tracker (figure 3.2) is as follows. The input video to each camera is first processed to extract all ball and player like candidates before running the object tracking module. The tracked objects from each view are then registered onto a model of the soccer field via the homography transformation for analysis in 3D. The 3D analytics module handles object correspondence and height estimation for the ball. Our approach is thus designed for multi camera systems for complete coverage of the soccer field. We begin with the problem of object detection.

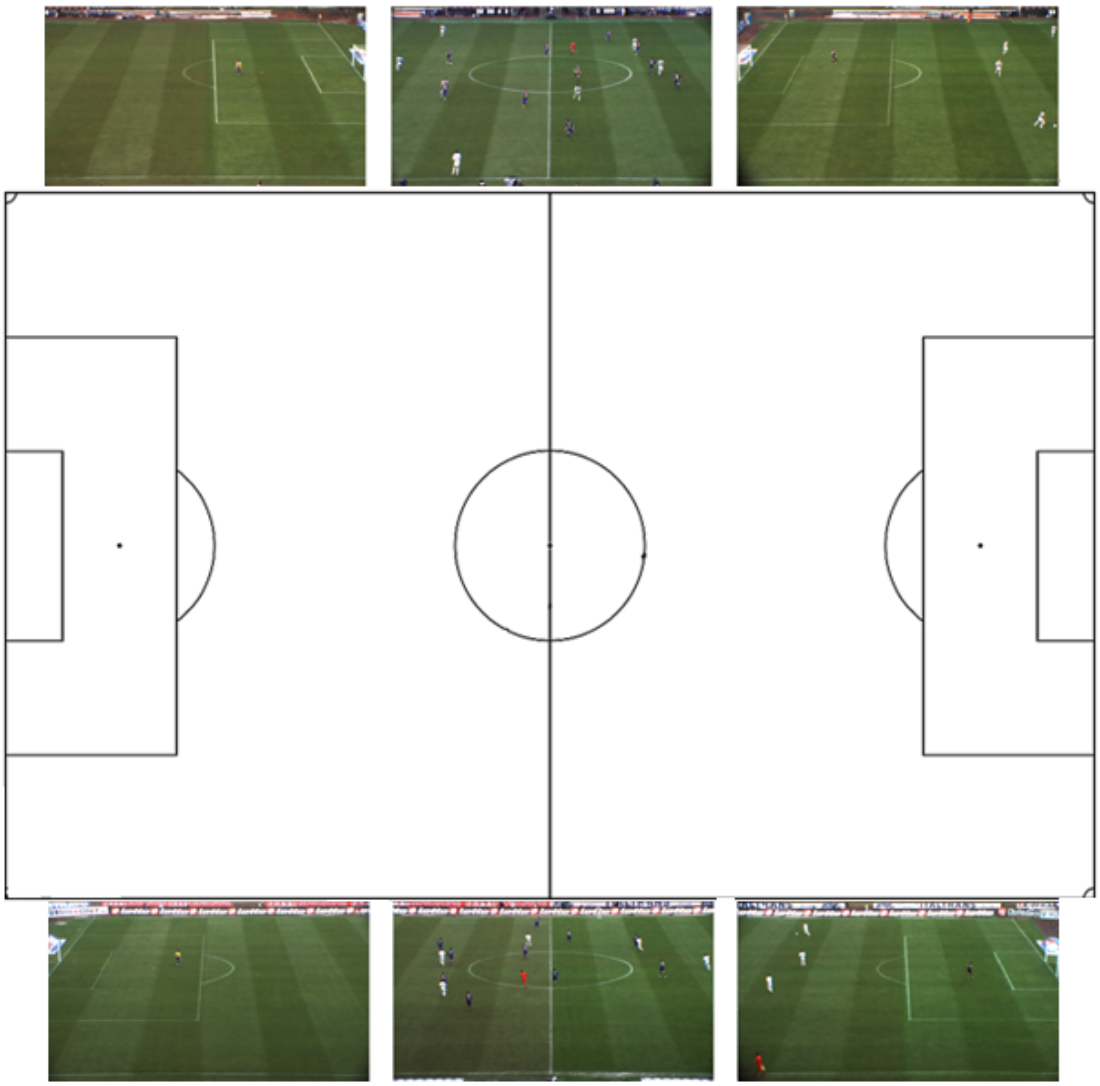

Figure 3.1: Camera placement and view for the ISSIA dataset

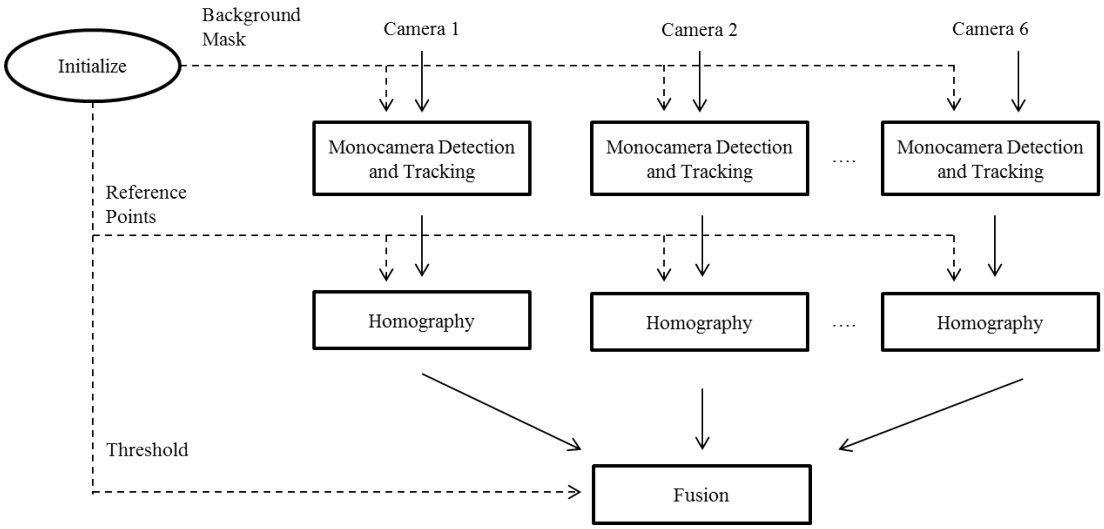

Figure 3.2: Framework for the multi-camera soccer tracker

Chapter 4

Object Detection

Every tracking problem requires an object detector as a precursor. In fact, the ability of the system to detect the desired object and reject false alarms is pivotal to the success of the tracker. In this chapter, we will provide a detailed description of our object detection framework. The first section covers our background subtraction algorithm, aimed to eliminate all pixels that are non-green in colour. As we will see, the result of the subtraction algorithm produces a binary image whose pixels need to be grouped into separate objects (section 4.2). Then from section 4.3 onwards, we will classify all these objects as either a ball or a player through various techniques.

4.1 Background Subtraction

Given an image, we want to identify regions containing the object of interest. Background subtraction is an image processing technique wherein an image's regions of interests (foreground) are extracted for further processing. The approach covered here takes advantage of prior knowledge that an image of the soccer field will be predominantly green in colour. More specifically, the histogram of the green channel of such an image is expected to contain multiple strong spikes. This approach thus aims to label every pixel as either a foreground or background depending on the value of its green channel. Given an input image, we first generate a histogram of its green channel with 256 bins.

Through the histogram (figure 4.1), the dominant pixel intensities which constitute the soccer field become noticeable. We then generate a binary mask of the input image via back-projection i.e. at each location (u,v) of the input image, we collect the value from the selected channel (green in this case), find the

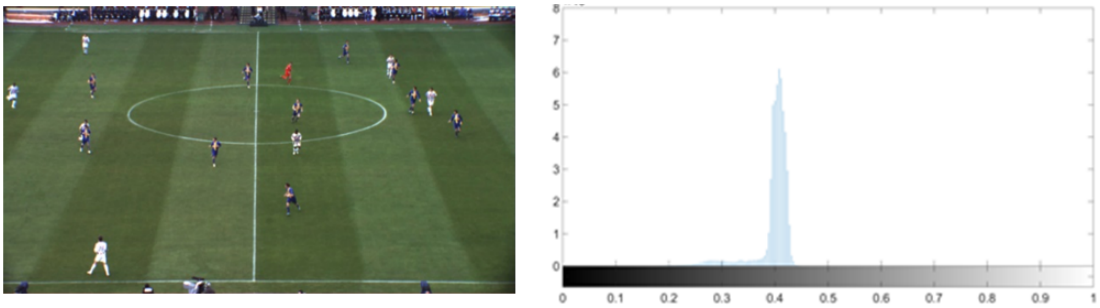

Figure 4.1: Input frame (left) and histogram of green channel (right)

corresponding histogram bin and either label it as a background or foreground by setting the mask at location (u,v) to 0 or 1 respectively. We label all pixels whose number of occurrence lies within 50% of the peak of the distribution as background pixels. All other pixels are labelled as foreground.

Figure 4.2: Result of background subtraction

4.2 Connected Components Labelling

As the current output is just a binary image filled with 1s and 0s, the next step involves extracting and labelling the various disjoint and connected components in an image, allowing for further analysis and is central to many automated image analysis applications.

We use an approach called connected components labelling. For this, we label the foreground pixels as a single entity if they are 8-connected. By definition, a set of pixels, P, is a connected component if, for every pair of pixels pi and pj $\subseteq$ P, there exists a sequence of pixels pi, . . . , pj such that all pixels in the sequence are 1 and every 2 pixels are adjacent. Figure 4.3 illustrates the output where the number on the object indicates its assigned identity.

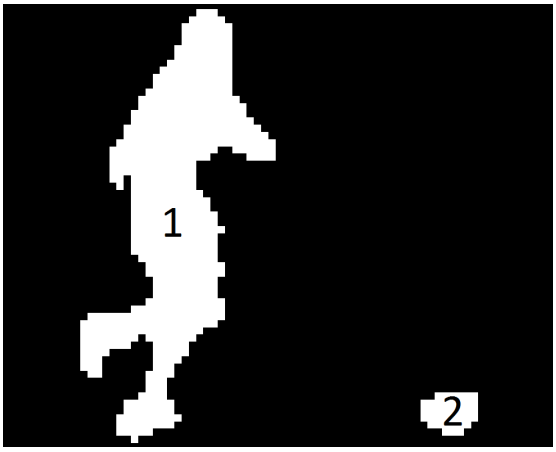

Figure 4.3: Connected components labelling

Table 4.1: Assigning objects via contour analysis

Contour	Roundness	Area	Object Assigned
	4.98	569.5	Player Candidate
	1.24	28.5	Ball Candidate

4.3 Contour Analysis

Given these components, including noise, the next task is to distinguish the different objects (ball or player). For this, we utilize object properties such as size and roundness. An objects roundness is determined from equation 4.1.

$$\text{Roundness} = \frac{\text{Perimeter}^2}{4\pi\text{Area}} \tag{4.1}$$

An object is labelled as a ball candidate if its roundness metric is within 0.5 1.5 and exhibits certain minimum area. For illustration, sample contours with their corresponding roundness and area are shown in table 4.1.

4.4 Automatic Player-Team Identification

Given the player labelled objects (including noise), our next task is to identify the team (cluster) he belongs to (Blue/White team or Referee). To this end, we have employed the K-means clustering algorithm to learn a set of features that describe the mean of all 3 clusters. Assuming we have a set of player labelled objects accumulated from each camera with feature vector x, K-means aims to segregate the data into k clusters by minimizing the within cluster sum of squares and is defined as follows.

$$argmin \sum_{i=1}^{k} \sum_{x \subseteq S_i} \|x - \mu_i\|^2 \tag{4.2}$$

Our feature vector is generated as follows. First, we crop a rectangular region around the player before resizing it to a resolution of 20 x 45. The images are then separated into its individual RGB channel before being concatenated to form a 2700 dimensional vector (20 x 45 x 3). Thus, given a total of N points (images) of the players and referees in a 2700 dimensional space, we partition the points into 3 clusters for the referee and 2 teams. This process is only run once upon initialization. Players in the subsequent frames are then assigned into their respective clusters via the nearest neighbour classifier.

4.5 Template Matching for Ball Filtering

As there can be several ball-labelled components returned from 4.2 and 4.3 due to noise and artefacts, we employ the template matching framework via normalized cross correlation (NCC) with an offline generated template as the final means to identify the best ball candidate. In the template matching framework, the template is correlated with the image with the goal of identifying similar patches in the image. At the image location (u, v), the normalized cross correlation is defined as

$$R(u,v) = \frac{\sum_{x',y'} T(x',y') \cdot I(u+x', v+y')}{\sqrt[2]{\sum_{x',y'} T(x',y')^2 \cdot I(u+x', v+y')^2}} \tag{4.3}$$

Where an output of 1 indicates a perfect match and -1 the worst match. We have manually generated 2 templates of the ball to be used interchangeably depending on the location of the object in the frame. A figure illustrating the output of the NCC is shown in figure 4.5.

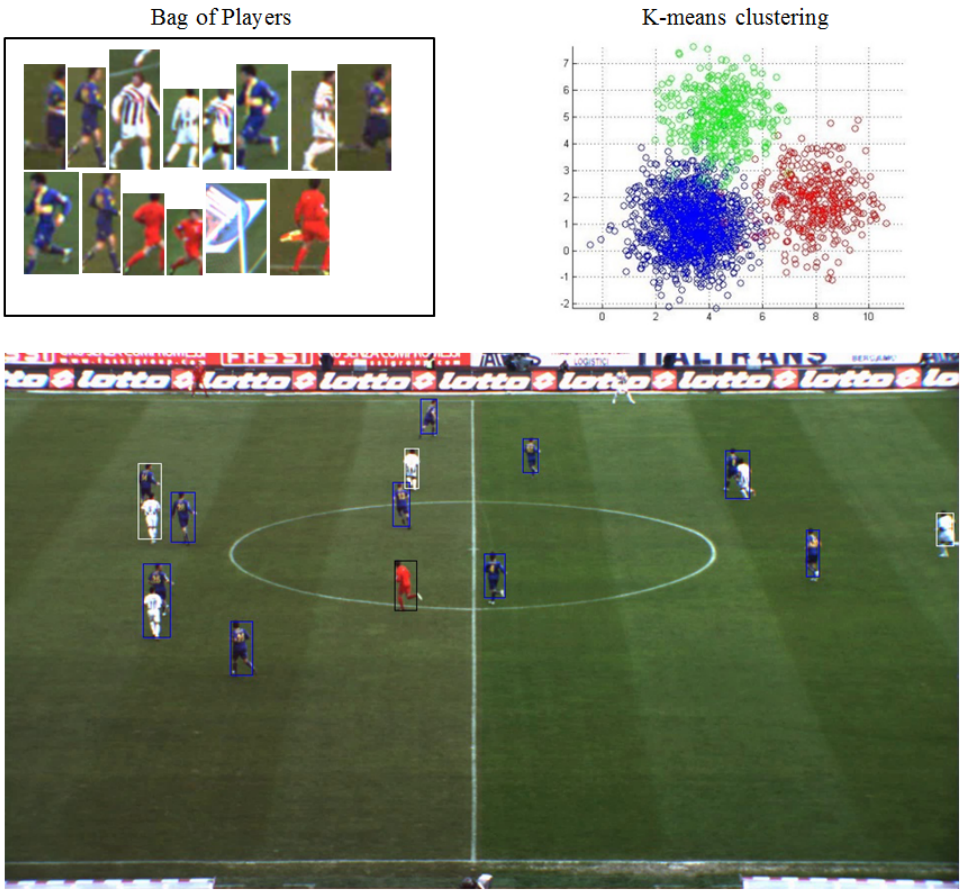

Figure 4.4: Player-Team Identification via K-means clustering

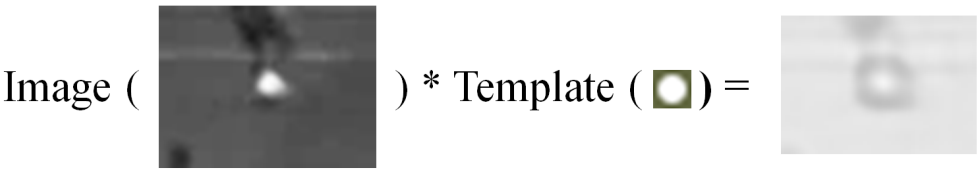

Figure 4.5: Template Matching via Normalized Cross Correlation

Before we end of the chapter on object detection, note that framework described for the ball detection up till now is not robust against noise as it is highly likely that the camera assigns any object as a ball if its area and roundness satisfies the threshold. In addition, the system treats each of the 6 input videos separately and will thus continue to search for the ball even if it has already been detected. These issues however, will be resolved in chapter 6 on multiple camera analysis when we integrate the information from multiple cameras. With that, we conclude the chapter on object detection.

Chapter 5

Object Tracking

Throughout the video, after a successful detection, we would like to establish the trajectory of the objects as it travels over consecutive frames. In this chapter, we discuss the tracking algorithms for both the ball and the players. Section 5.1 will discuss the ball tracking algorithm while 5.2 the player tracking algorithm.

5.1 Ball Tracking

Due to the unpredictable motion of the ball, we implement a simple tracking algorithm utilizing prior knowledge of soccer that works in conjunction with the player tracker for improved accuracy. The technique employs the template matching framework of section 4.5 with the only difference being where and how the search is performed would depend on the status of the ball. In the following sections, we go through the different scenarios that occur during a soccer match and explain the minor changes in the template matching framework that we use to successfully track the ball throughout the video.

5.1.1 Ball in Play

Suppose a ball has been detected, we construct a window, manually designed to be large enough such that it contains the ball at the next frame even at its maximum speed. We update two variables for every frame. A tracking lifetime metric, which is incremented for every successful track and a ball lifetime metric, which is incremented for every frame after a successful detection. The usage of these variables will be explained in the next section.

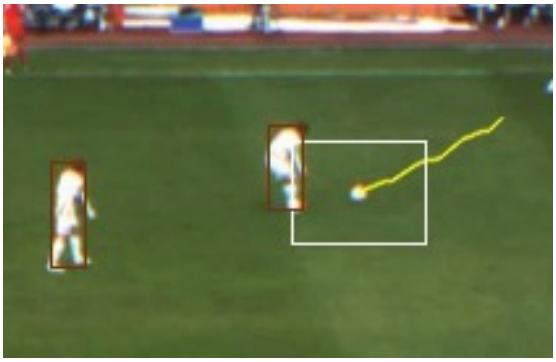

Figure 5.1: Default window size 45 x 35

5.1.2 Player-Ball Occlusion

A common scenario that occurs in soccer is when the ball goes under occlusion as the player dribbles it. To handle such types of occlusion, when the distance of the ball to the nearest player reaches below a threshold (15 pixels in our implementation), we construct a larger window (100x60) and attach the center at the feet of the player and assume that the player is dribbling the ball. Then, for all other bounding boxes (players) enveloped within that window, a mask is generated so as to prevent misdetection.

For every frame it remains attached to a player, however, we do not increment the tracking lifetime by 1. This to prevent scenarios in which the system fails to locate the ball as it leaves the player and hence tracks the player for an indefinite duration. The track ratio, defined below, ensures that the window will remain attached to the player's feet until a match has been found or its track ratio reaches below a threshold.

$$\text{Track Ratio} = \frac{\text{tracking lifetime}}{\text{ball lifetime}} \tag{5.1}$$

Figure 5.2 shows the system tracking the ball under player-ball occlusion till it escapes.

5.1.3 Ball out of play

When the ball goes out of bounds, a throw-in is awarded to the opposing team. During the throw-in, up till the ball enters the field of play, the player tasked to throw the ball in is the only player allowed to stand beyond the side-line. Therefore, when such an event occurs, we restrict the detector to perform its search only on the regions and the player whose position lies outside the boundaries of

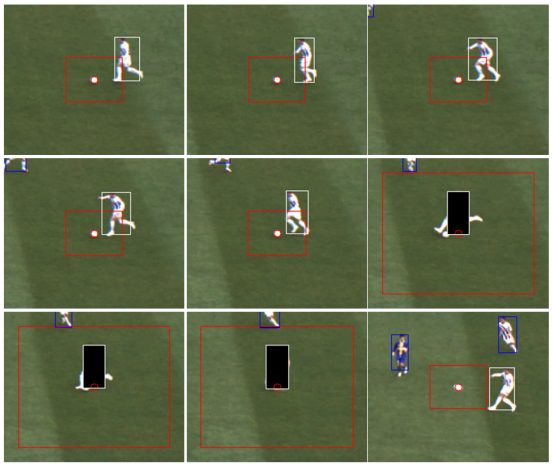

Figure 5.2: The frames 62, 64, 66, 68, 72, 74, 76, 68, 80 are shown

the soccer field.

In addition, a slight change to the algorithm described in 5.1.2 if the ball is located within the bounding box of the player is that no mask will be generated. Instead, we will assume the location of the ball to be at the top of the bounding box if no detections are found. This is akin to assuming that the player is holding onto the ball, about to throw it towards his team-mates. As the ball re-enters the field of play, the tracker will revert back to either the algorithm described in 5.1.1 or 5.1.2, depending on the status of the ball (Figure 5.3).

Figure 5.3: The frames 52, 58, 83, 108, 115, 125, 134, 190, 195 are shown

Note that the algorithm described above only applies to the side-line situated closer to the camera as it is much more challenging to detect a ball going out of play for the side-line at the opposite end. The placement of the cameras however, ensures that the side-lines for both sides of the field are covered.

5.2 Player Tracking

The player tracker is based on a Kalman filtering framework which uses the equations of kinematics while accounting for statistical variations in the measurements and the model itself. We will simply state the Kalman filtering algorithm without delving into the derivation of the equations.

5.2.1 Kalman Filter

The algorithm consists of two stages, a predict stage which uses the learned motion model, the past state to estimate the next state, and an update stage which uses the measurements corresponding to the next state to correct the motion model as well as to give a better estimate of the state vector given the prediction and measurements (figure 5.4). In our implementation, the measurements for the update stage are given by the centroid of the nearest player (output of connected components analysis) if the distance is below 20 pixels. The update stage will not be invoked if the distance condition is not satisfied i.e. the player will take on its predicted coordinate.

5.2.2 Occlusion Handling

In figure 5.5, a single bounding box is generated for multiple players if they lie in close proximity. This is the result of the background subtraction and connected components algorithm returning a single connected object consisting of 2 players. We resolve occlusion by taking advantage of the fact that in soccer videos, occlusions only occur when players move past each other. More specifically, each bounding box can only contain the coordinate of a single player; any more would indicate the presence of occlusion. Our occlusion handling algorithm is thus designed to segment players under occlusion via template matching. Given multiple players under occlusion, we employ a template matching framework using an image from the previous frame. For each bounding box in question, we convolve with it a template. The template with a higher score will then be selected to mask out the region. This process is repeated until all occluded players are segmented (figure 5.6)

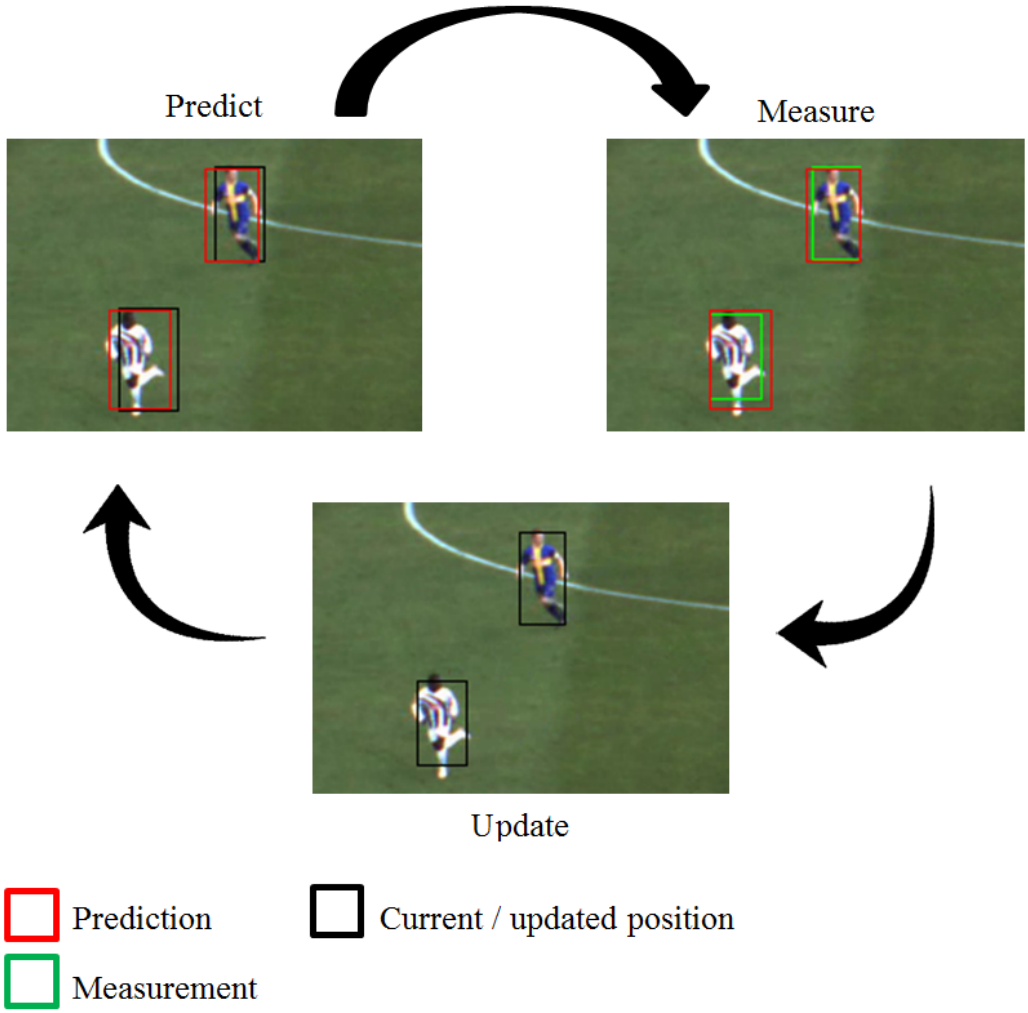

Figure 5.4: Kalman filtering framework for the player tracker

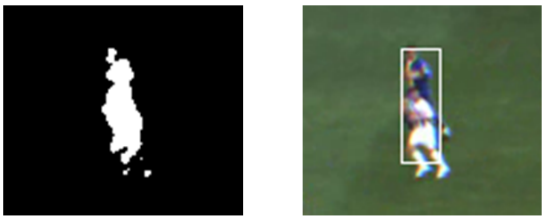

Figure 5.5: Occlusion in player tracking

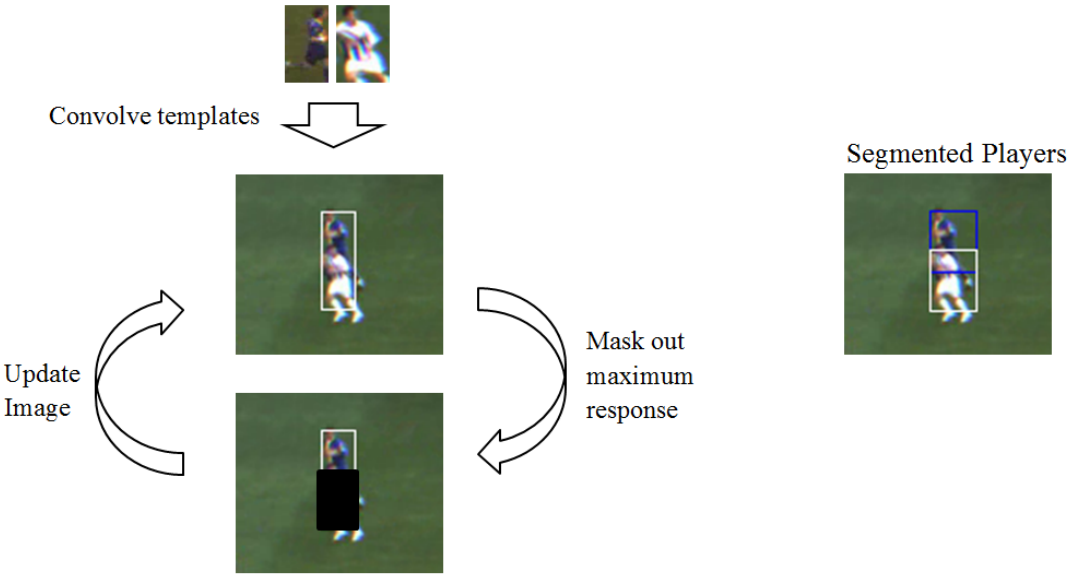

Figure 5.6: Occlusion handling framework

Chapter 6

Multi Camera Analysis

In the previous chapters, we were able to detect and track the objects in 2D. The location of the objects are however, in terms of image pixels. Also, recall that the system detects and tracks the best ball candidate (if there is one) in each view. Hence at any point of time, we can have up to 6 tracked balls in the field. What we would like to achieve is the ability to identify candidates that correspond to the real ball. Furthermore, since the ball often flies across the air, we would like to have a good estimate of its actual position in 3D.

In this chapter, we will finally integrate the information from all the cameras to present the results in terms of world coordinates. We will start with a theoretical derivation of the homography matrix in section 6.1. This matrix relates points between 2 image planes and is responsible for mapping the players and the ball from each camera view onto a common model of the soccer field. The problem of establishing correspondence between detected objects across cameras then follows in sections 6.2 and 6.3, i.e., given the projections of all players on the field model, we need to identify the pairs that belong to the same object in the real world. In the case of the ball, we need to identify the pair that corresponds to the real ball. In section 6.4, we present the algorithms used to localize the ball in 3D.

6.1 The Homography Matrix

When light hits an object, it is either absorbed or reflected back into the scene. Some of this reflected light travels through the pinhole, forming a 2D image on the plane. A general mathematical expression describing the relationship between a 3D world point and its 2D image point is known as the camera matrix.

$$\begin{bmatrix} wu \\ wv \\ w \end{bmatrix} = \begin{bmatrix} f & s & u_o \\ 0 & f & v_o \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \tag{6.1}$$

Where the 3x3 matrix is referred to as the camera intrinsic matrix and the 3x4 matrix the extrinsic matrix. If we assume the scene to be planar, i.e. all world points lie on a plane, then, without any loss of generality, we can re-define the world points as $X = [x\ y\ 0]^T$ and the camera matrix thus reduces to the following expression.

$$\begin{bmatrix} wu \\ wv \\ w \end{bmatrix} = \begin{bmatrix} f & s & u_o \\ 0 & f & v_o \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & t_x \\ r_{21} & r_{22} & t_y \\ r_{31} & r_{32} & t_z \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \tag{6.2}$$

By multiplying out the constants, we obtain the resultant 3x3 matrix, denoted as H

$$\begin{aligned} \begin{bmatrix} wu \\ wv \\ w \end{bmatrix} &= \begin{bmatrix} f & s & u_o \\ 0 & f & v_o \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & t_x \\ r_{21} & r_{22} & t_y \\ r_{31} & r_{32} & t_z \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \\ \begin{bmatrix} wu \\ wv \\ w \end{bmatrix} &= \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \end{aligned} \tag{6.3}$$

This is known as homography, or the planar projective transformation where (h11,,h33) represent the parameters that describe the linear transformation between 2 planes. The homography matrix is invertible in a sense that a point projected onto another plane can be re-projected back onto the original plane. If we expand and manipulate equation 6.3, we obtain an under-constrained system of equations 6.4. We can thus estimate H by establishing the correspondence between at least 4 sets of image and world points through the knowledge of several features such as the corners of the soccer field to get the resultant over-constraint expression 6.5.

$$\begin{bmatrix} x_1 & y_1 & 1 & 0 & 0 & 0 & x_1u_1 & y_1u_1 \\ 0 & 0 & 0 & x_1 & y_1 & 1 & x_1v_1 & y_1v_1 \end{bmatrix} \begin{bmatrix} h_{11} \\ h_{12} \\ h_{13} \\ h_{21} \\ h_{22} \\ h_{23} \\ h_{31} \\ h_{32} \end{bmatrix} = \begin{bmatrix} u_1 \\ v_1 \end{bmatrix} \tag{6.4}$$

$$\begin{bmatrix} x_1 & y_1 & 1 & 0 & 0 & 0 & x_1u_1 & y_1u_1 \\ 0 & 0 & 0 & x_1 & y_1 & 1 & x_1v_1 & y_1v_1 \\ & & & & \vdots & & & \\ 0 & 0 & 0 & x_1 & y_1 & 1 & x_nv_n & y_nv_n \end{bmatrix} \begin{bmatrix} h_{11} \\ h_{12} \\ h_{13} \\ h_{21} \\ h_{22} \\ h_{23} \\ h_{31} \\ h_{32} \end{bmatrix} = \begin{bmatrix} u_1 \\ v_1 \\ \vdots \\ u_n \\ v_n \end{bmatrix} \tag{6.5}$$

This is of the form A$\mathbf{H}$ = b which can be solved to yield to solution vector

$$\mathbf{H} = (A^T A)^{-1} A^T b \tag{6.6}$$

Thus the linear transformation between the 2 image planes have been derived. The data used for homography estimation has been included under Appendix A.

6.2 Object Registration

With the image coordinates of a ball and the players and the homography matrix all cameras, we project all objects onto the field model, shown in figure 6.1. The colour of the circle indicates the players team while the number inside the circle the camera that was tracking it. The ball candidates are indicated by a black circle with a red outline and are being tracked by cameras 1 and 3 as indicated. Notice the presence of noise on being projected near the left goalpost. As mentioned at the end of chapter 4, such results occur due to the inaccuracy of the detection framework.

Since the dimensions of the model are proportional to the actual field, we can apply a scaling operation to convert the coordinates of all projections from pixels

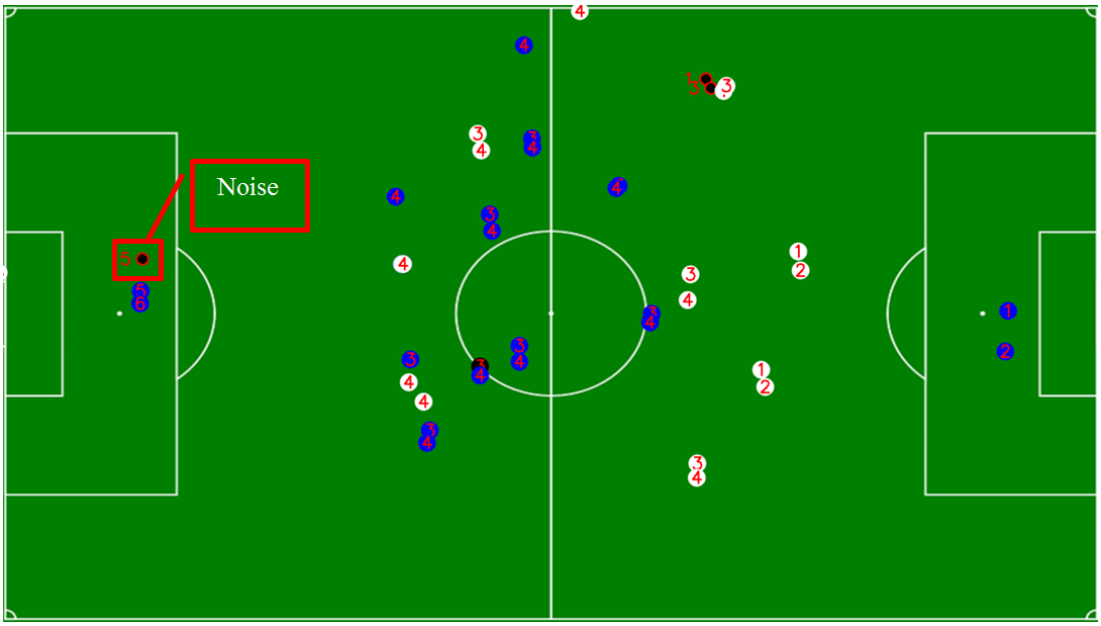

Figure 6.1: Registering all objects onto the field model (Frame 339)

to meters.

6.3 Fusion from Multiple Cameras

As seen in the previous section, the effect of registering the objects from multiple cameras produces multiple objects on the field model. The next step would then be to establish correspondence between all pairs of players and the ball.

6.3.1 Player Fusion

The nearest neighbour method is used for determining correspondence between pairs of players i.e., for each player i, we compute the Euclidean distance between every other player j from the same team and declare the pair with the smallest Euclidean distance as the same object if their distance does not exceed a certain threshold (equation 6.7). The result is shown in figure 6.2.

$$\text{argmin}\|p_i - p_j\| < \text{threshold} \tag{6.7}$$

Players i who do not meet the criteria above will still remain in the field albeit without a pair. In this case, it is assumed that the player is being tracked by a single camera and thus does not have a closest pair. Such a scenario also occurs when the occlusion handling mechanism fails.

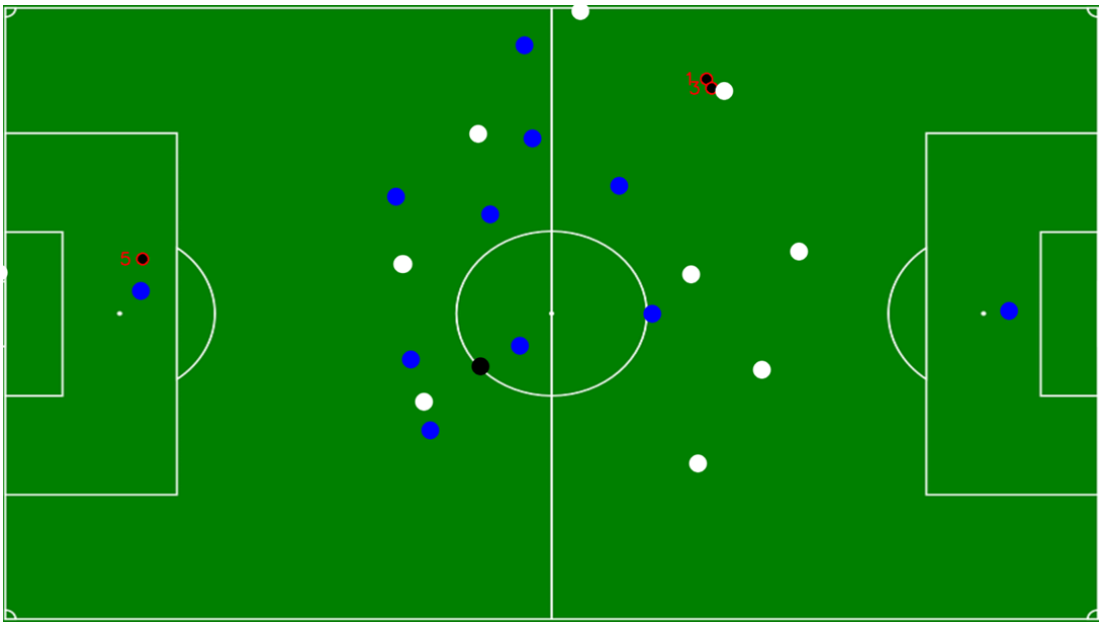

Figure 6.2: Player Correspondence (Frame 339)

6.3.2 Ball Fusion via Epipolar Constraints

A different method must be used for the ball as it can be easily observed in figure 6.3 (where ball 5 and 6 belongs to the same object) that the nearest neighbour method would not work. Such a result can be reasoned by the fact that the homography transformation assumes all points in the scene lie on a plane. The image coordinate of a ball flying across the air would thus be higher. It is extremely important to establish the correct correspondence for the ball as identifying a false candidate would ruin localization and the remaining sections on localization in 3D. In order to achieve proper correspondence, we employ the epipolar geometry of stereo vision in which a point viewed by 2 cameras generates a set of geometric constraints that relates their 2D position in both cameras [13].

With reference to figure 6.4, the Epipolar geometry can be explained as follows. Suppose that a point in the real world (denoted by the red circle), observed by 2 cameras and is projected onto the field model (denoted as and respectively) via the homography transformation. Then, the back-projected lines and will always intersect at the location of the object in 3D. It is this property that is of most significance in searching for a correspondence.

These lines however, do not intersect in practice due to errors caused by camera calibration so instead, we will iterate through all possible pairs and compute the

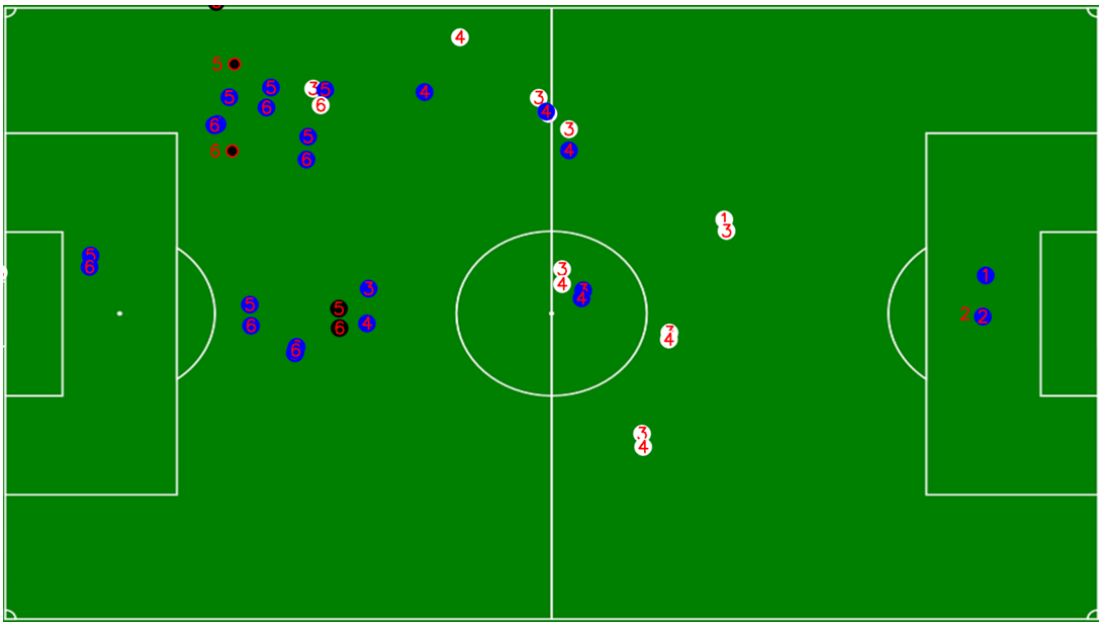

Figure 6.3: Registering all objects onto the field model (Frame 433)

shortest distance between the skewed lines to determine if they belong to the same object, namely the ball. Note that the method used to extract the camera coordinates follows [14] with the exception of the usage of maximum likelihood estimation to minimize the re-projection error. Appendix C lists the data and results of camera calibration.

6.4 Height Estimation

Upon designating the pair of projections as the true ball, we can localize its position in 3D by assigning the midpoint of the perpendicular line as the estimated 3D position of the ball, assuming that the errors from both cameras are equal in magnitude. In figure 6.6, where c denotes the camera location, b the projection of the ball onto the field model, the height of the ball is assumed to be b.

When the number of detections is reduced from 2 to 1, cues about the 3D position of the ball can be obtained by establishing a plane from last known 3D positions. In this scenario, it is assumed that the trajectory of the ball travels along a vertical plane, perpendicular to the ground plane (Figure 6.7).

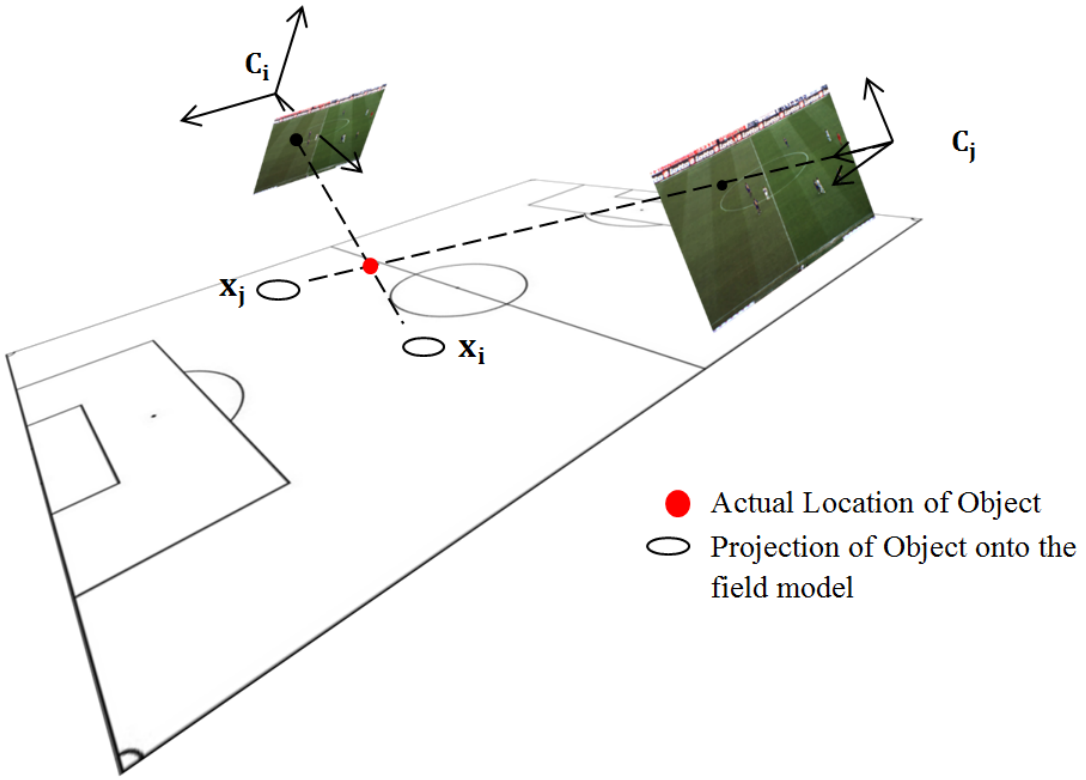

Figure 6.4: An illustration of the Epipolar geometry in multi view soccer videos.

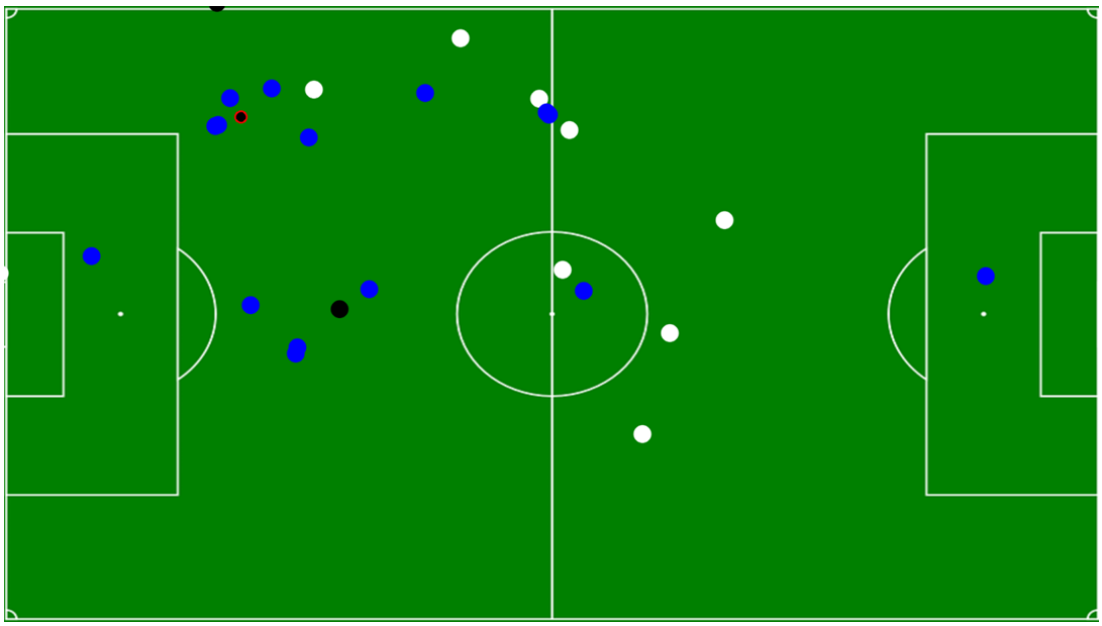

Figure 6.5: An illustration of the Epipolar geometry in multi view soccer videos.

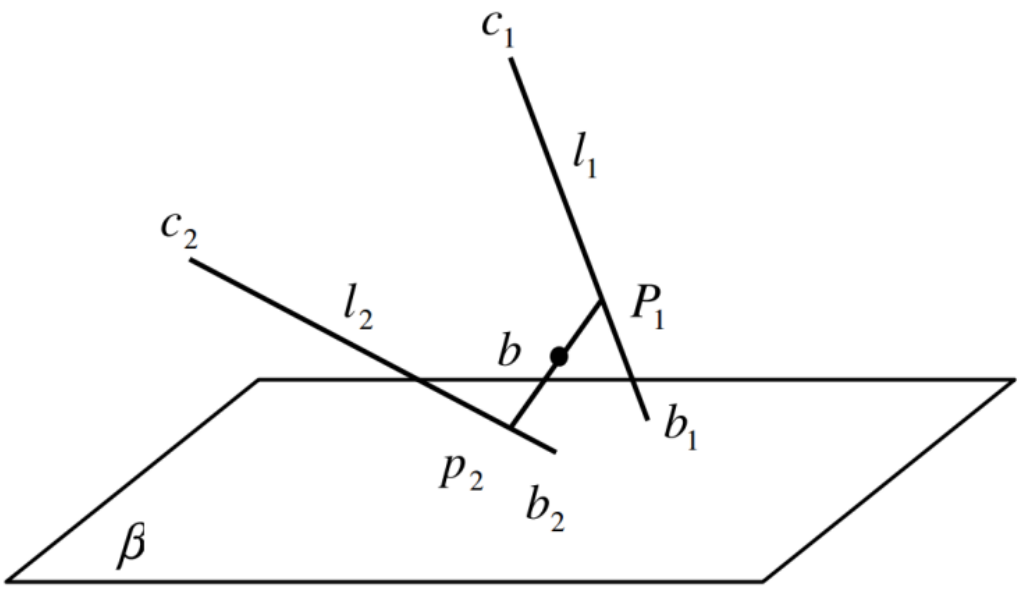

Figure 6.6: Estimating the 3D position of the ball using multiple cameras

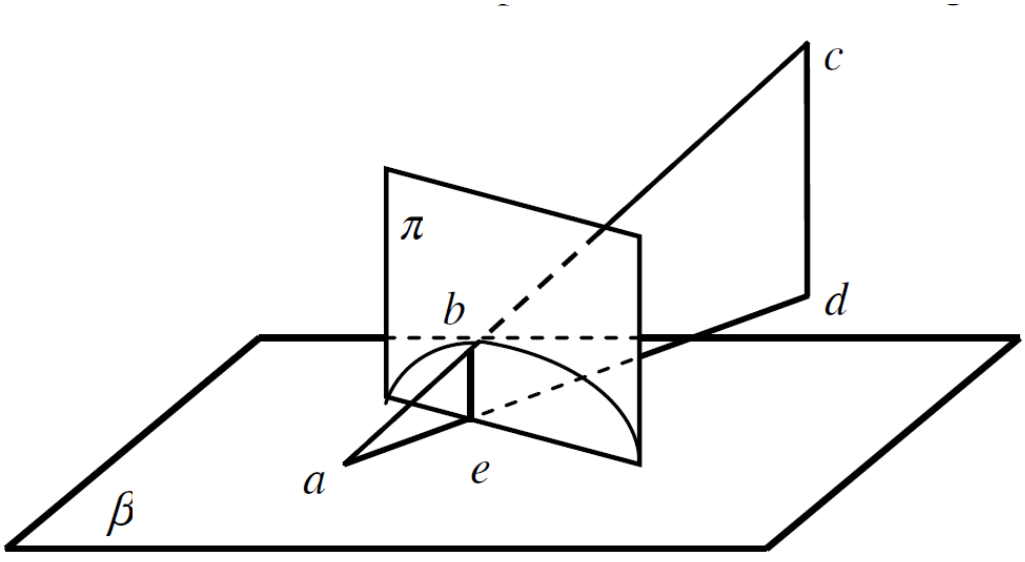

Figure 6.7: Estimating the 3D position of the ball using a single cameras

Figure 6.8: Height Estimation. The frames 610, 615, 620, 625 of camera 5 from top left to bottom right are shown. The ball is out camera 5s field of view for frame 620 and 625.

Figure 6.9: Height Estimation. The frames 610, 615, 620, 625 of camera 6 from top left to bottom right are shown. The height of the ball in frame 620 and 625 is estimated using a single camera.

Chapter 7

Results

An important aspect of tracking algorithms is its performance. Being able to assess the reliability of a tracking system via defined metrics provides a useful way to gauge its performance against the state of the art algorithms. More importantly, quantitative analysis of the designed tracker helps measure progress as it provides feedback for future improvements. In this chapter, we present the results of the ball tracking algorithm and discuss some of the limitations of the system.

7.1 Results for Ball Tracking

For tracking in 2D, the tracker was evaluated based on the commonly used precision and recall scores. Their definitions, along with all relevant quantities are all defined below.

True Positive: Total number of frames where both ground truth and system results concur on the presence of the ball with the centroid of their bounding boxes lying within a specified distance.
False Positive: Number of frames where the ground truth does not contain the ball but the system detects an object.
True Negative: Total number of frames where both ground truth and system results concur on the absence of the ball.
False Negative: Number of frames where the ground truth contains the ball, while the system either does detect the ball or the detected ball does not lie within a specified distance.

For multi camera tracking, the performance was assessed by computing the cov-

erage of the ball, i.e. the percentage of frames for which the ball was tracked by at least 1 or 2 cameras and the triangulation error, i.e. the distance between the skewed lines. Figure 7.1 displays the classification of tracks for each camera over 3000 frames with their individual precision and recall scores tabulated in the table below. The true positive and false negatives are computed with an error threshold of 10 pixels. From the figure, it can be seen that cameras 2 and 3 each suffers from high false negative rate compared to other cameras.

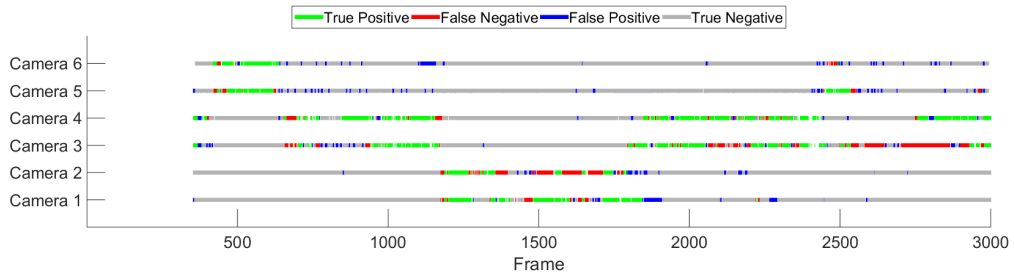

Figure 7.1: Classification of object trajectory over 3000 frames

This can be attributed to the fact that the ball for the ISSIA soccer video is under frequent occlusion with respect to said cameras. For example, figures 7.2 and 7.3 explain the high false negative rate for camera 3 from frame 2500 to 2800. From frame 2530 onwards, camera 3 fails to track the ball as it goes out of play. It also remains unseen by the opposite camera 4 until frame 2749 when a player appears in camera 4's field of view to throw the ball in. The distance of the ball from camera 3 and the surrounding region makes re-detection a challenging task.

Figure 7.2: The ball goes out of play in frame 2530 (left) of camera 3 and remains undetected. Frame 2749 is shown on the right

Despite the low precision and recall measures for cameras 2 and 3, the capability of the system in tracking the ball over 3000 frames is rather sufficient as the placement of cameras on opposite sides of the soccer field ensures that the ball remains visible most of the time. The results (figure 7.4 and table **??**) display the coverage of the soccer ball when taking all 6 cameras into consideration. The results are promising with the ball being tracked over 80% of the frames.

Figure 7.3: The ball goes out of play in frame 2530 and is only re-detected when it appears in camera 4s field of view in frame 2749

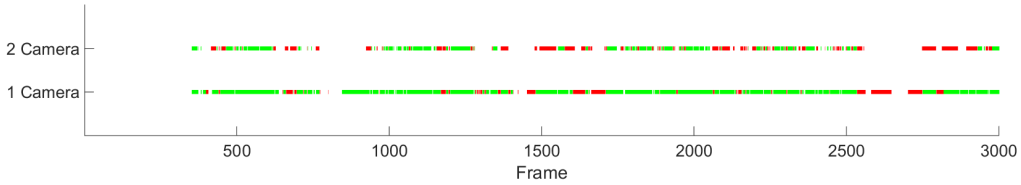

Figure 7.4: Ball coverage over 3000 frames. The top row indicates frames for which the ball is tracked by at least 1 camera while the bottom frames at which the ball is tracked by at least 2 cameras

Table 7.1: Ball coverage over 3000 frames

Ball Coverage	
By at least 1 camera	By at least 2 camera
1948 out of 2431	895 out of 1781

Lastly, the triangulation error over 3000 frames have been summarized by its minimum, maximum and average error. With an average error of 0.4 meters, the results are much better than the sensor based systems that rely on radio signals. To summarize, figures 7.5 and 7.6 indicate the ball tracking precision and recall scores with an error threshold from 0 to 30 meters. The plot in red indicate the performance of the tracker without the use of multiple cameras to assist the detector and in blue the tracker incorporating 3D information. With an error threshold of 10 pixels (table 7.1) a difference of 18.2 and 9.8 percentage points in both precision and recall scores indicate the importance of the 3D analytics framework. By localizing the ball in 3D, we were able to activate / deactivate the ball tracker for each camera depending on the location of the ball, thereby reducing the false positive rate.

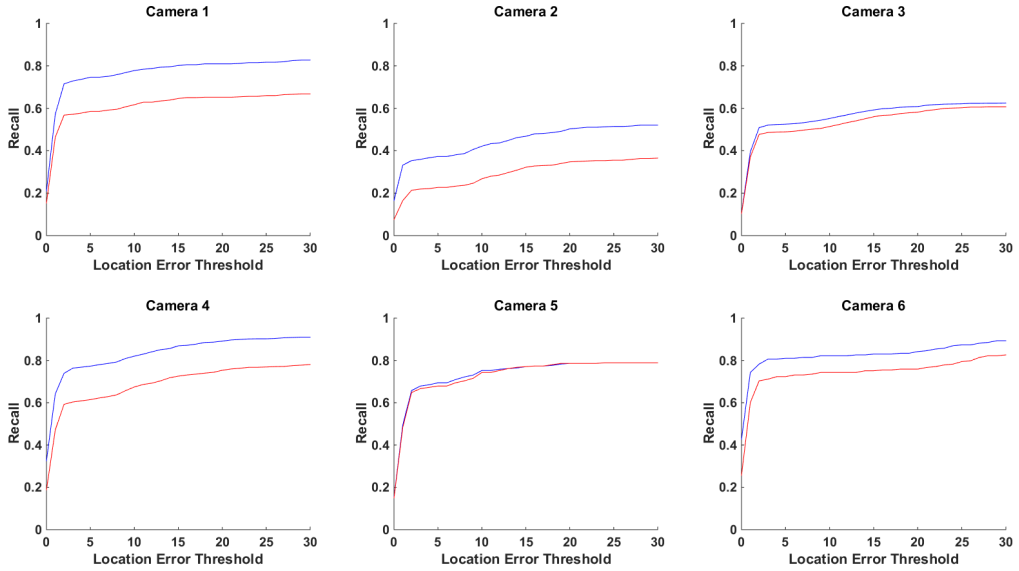

Figure 7.5: Recall plots for each camera

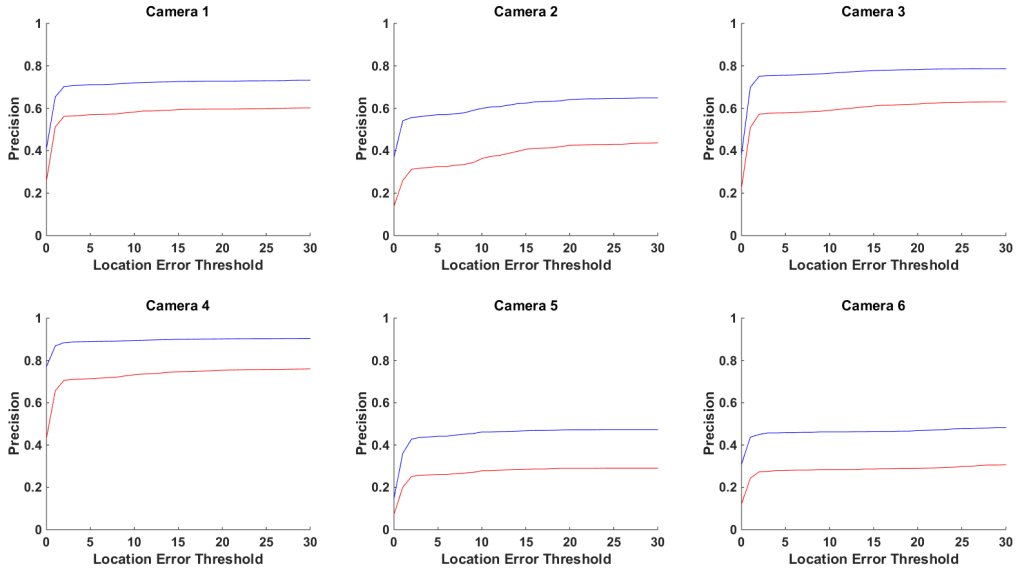

Figure 7.6: Precision plots for each camera

7.2 Limitation of the Occlusion Handling Algorithm

The occlusion handling algorithm described in section 5.2.2 is only invoked when a bounding box contains the coordinates of at least 2 players. As such, it is unable to segment players who are occluded at the start of the video as their pixels will be

connected and thus declared as a single object. This is shown in figure 7.7 when the tracker was initialized at frame 340. As can be seen, the occlusion handling algorithm fails to segment the pair of players who were already 'connected' at the beginning.

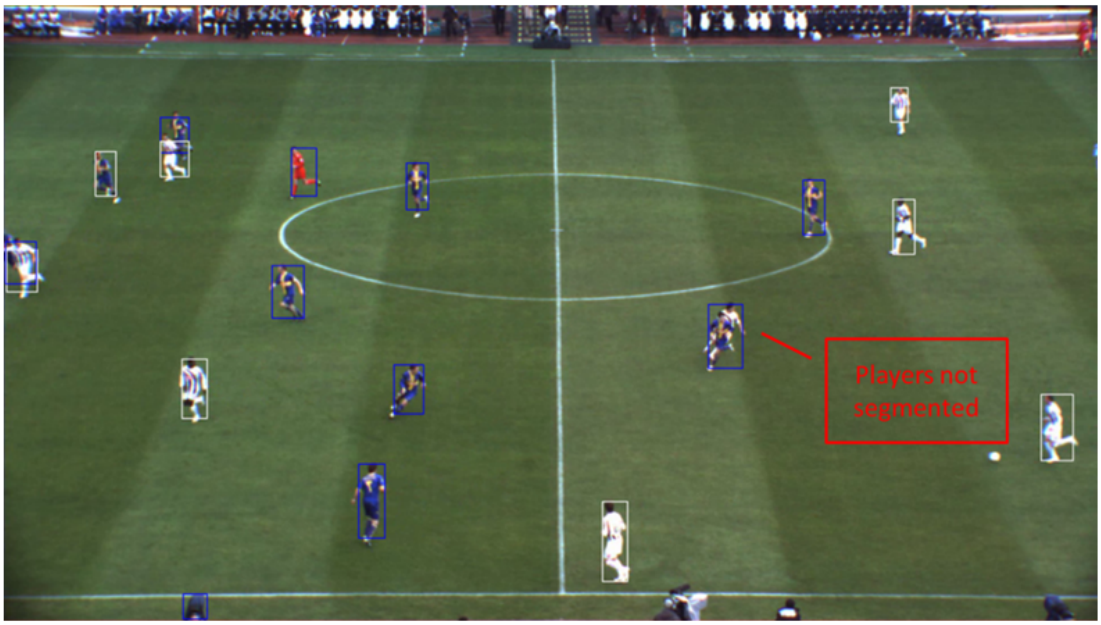

Figure 7.7: Frame 355 of Camera 4. The players highlighted by the red box are not segmented

7.3 Limitation of K-Means for Team Identification

As was previously indicated in section 4.4, the features that describe the 2 teams and the referee are generated via K-Means clustering upon initialization. The players are then assigned into their respective teams in subsequent frames based on the Euclidean distance to the generated cluster centers. An inherent limitation of the traditional K-Means for clustering is that the generated centers are affected by the composition and structure of the data. As there is an in imbalance of class samples as the players on each team far outnumber the referees, the cluster center for the referee may not be in the 'ideal' position. Additionally, the method used to generate the feature vector plays an important role in the success of the team identification algorithm. In figure 7.8, the referee and some of the players have been assigned a wrong label.

Figure 7.8: Frame 2325 of Camera 4. The referee and some of the players have been labelled wrongly

Chapter 8

Conclusion

An automated real time tracking system for soccer videos has been presented in this thesis. Its distinction between other existing vision based methods for ball tracking is (1) the utilization of Epipolar constraints for fast object detection, (2) the use of prior knowledge to handle game situations and (3) the coordination with the player tracker for occlusion handling. Experiments on the ISSIA soccer dataset verify the efficacy of the proposed methods with the potential to do even better. Although the precision and recall measures of the prototype are not as competitive as the state of the art trackers, the simplicity of the designed algorithms allows ease of integration with other state of the art trackers. We therefore conclude that the system has the potential to outperform other trackers.

Possible improvements would be to change the subtraction algorithm as it was noted that the algorithm does not work well on a different dataset with varied illumination. The weakness of the current approach for ball tracking is the usage of templates for ball detection. Extensions could therefore be the integration of a trajectory analysis based algorithm to learn and generate a template online which could then be reused for the future soccer matches provided the camera setup is similar. An algorithm that extrapolates missing or eliminates false trajectories could also be explored. Additionally, tracking of the ball via physics would be helpful if the camera calibration errors are miniscule. It was not successfully done in this thesis due to said errors.

For player tracking, a robust correspondence and occlusion handling algorithm that takes into account the total number of players in the field would work well provided it is able to handle a reduction in the number of players (red card). Additionally, the limitation of the automatic player team identification described

in section 4.4 is its inability to identify the separate goalkeepers. Furthermore, the performance of the clustering algorithm is suboptimal due to the fact that the input to the clustering algorithm comes from every single camera. As such, there will be more player samples than referees or goalkeepers.

Bibliography

[1] Y. Ohno, J. Miura, and Y. Shirai. Tracking players and estimation of the 3d position of a ball in soccer games. In *IEEE International Conference on Pattern Recognition*, 2000.

[2] J. Ren, J. Orwell, G. Jones, and M. Xu. Real-time modelling of 3-d soccer ball trajectories from multiple fixed cameras. In *IEEE Transaction on Circuits and Systems for Video Technology*, 2008.

[3] T. Kim, Y. Seo, and K. S. Hong. Physics-based 3d position analysis of a soccer ball from monocular image sequences. In *IEEE International Conference on Computer Vision*, 1998.

[4] I. Reid and A. North. 3d trajectories from a single viewpoint using shadows. In *British Machine Vision Conference*, 1998.

[5] K. Matsumoto, S. Sudo, H. Saito, and S. Ozawa. Optimized camera viewpoint determination system for soccer game broadcasting. In *International Association for Pattern Recognition*, 2000.

[6] X. Yu, Q. Tian, and K.W. Wan. A novel ball detection framework for real soccer video. In *IEEE International Conference on Multimedia and Expo*, 2003.

[7] X. Yu, C. Xu, Q. Tian, and H.W. Leong. A ball tracking framework for broadcast soccer video. In *IEEE International Conference on Multimedia and Expo*, 2003.

[8] T. Shimawaki, T. Sakiyama, and Y. Shirai J. Miura. Estimation of ball route under overlapping with players and lines in soccer video image sequence. In *IEEE International Conference on Pattern Recognition*, 2003.

[9] K. Choi and Y. Seo. Tracking soccer ball in tv broadcast video. In *Image Analysis and Processing*, 2005.

[10] V. Pallavi, J. Mukherjee, A.K. Majumdar, and S. Sural. Ball detection from broadcast soccer videos using static and dynamic features. In *Journal of Visual Communication and Image Representation*, 2008.

[11] T. D'Orazio and M. Leo. A review of vision-based systems for soccer video analysis. In *Pattern Recognition*, 2010.

[12] T. DOrazio, M. Leo, N. Mosca, P. Spagnolo, and P. L. Mazzeo. A semi-automatic system for ground truth generation of soccer video sequences. In *IEEE International Conference on Advanced Video and Signal Surveillance*, 2009.

[13] Hartley, Richard, and Andrew Zissermano. Multiple view geometry in computer vision. 2nd edition. In *Cambridge: Cambridge University Press, 2004. Cambridge Books Online. Web. 07 November 2015.*

[14] Z. Zhang. A flexible new technique for camera calibration. In *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2000.

Appendix A
Camera Homography Data

Field Model Coordinates		Image (Camera 1) Coordinates	
1248	0	145	105
1440	0	570	110
1440	1080	200	1030
1620	220	1000	205
1620	860	1030	710
1820	400	1535	305
1820	625	1650	525
1824	1080	1870	1030
1920	0	1645	105
1920	220	1720	205
1920	400	1800	305

Field Model Coordinates		Image (Camera 2) Coordinates	
1440	0	1760	1040
1620	220	905	710
1620	860	930	200
1440	1080	1365	105
1344	1080	1570	100
1248	1080	1780	100
1720	540	635	400
1920	685	130	300
1920	1080	290	100

Camera Homography Data, Cont'd

Field Model Coordinates		Image (Camera 3) Coordinates	
768	0	515	95
795	545	455	400
770	1080	70	1025
960	0	940	95
960	395	943	300
960	685	944	516
960	1080	943	1025
1152	0	1370	95
1125	545	1430	400
1152	1080	1755	1025

Field Model Coordinates		Image (Camera 4) Coordinates	
1344	1080	120	95
1152	1080	544	95
1125	545	470	400
1152	0	85	1040
960	1080	956	95
960	685	951	296
960	395	946	516
960	0	940	1035
770	1080	1390	95

Camera Homography Data, Cont'd

Field Model Coordinates		Image (Camera 5) Coordinates	
0	0	275	95
0	220	195	200
0	400	115	300
100	400	385	300
100	685	260	525
96	1080	15	1035
300	220	930	195
300	860	895	710
480	0	1355	95
480	1080	1715	1035
672	0	1775	95

Field Model Coordinates		Image (Camera 6) Coordinates	
0	685	1800	300
0	860	1720	200
0	1080	1640	105
96	0	1880	1030
100	400	1655	520
100	685	1535	295
300	220	1030	700
300	860	1005	200
480	0	235	1030
480	1080	605	105
672	1080	195	105

Appendix B: Camera Calibration Data

			Camera 1		Camera 2	
	World Coordinates		Image Coordinates		Image Coordinates	
Giallo 1	77.261978	9.3193226	240	840	1405	165
Giallo 2	77.568573	34.337677	425	395	1495	400
Giallo 3	77.648048	57.3634	525	180	1650	775
Giallo 4	97.427101	56.906445	1375	180	290	765
Giallo 5	96.904655	9.5490179	1605	795	580	170
Fisso 6	104.86995	0.1017158	-	-	295	105
Fisso 7	104.90186	13.779555	-	-	210	205
Fisso 8	88.314507	15.439806	1025	705	930	205
Fisso 9	88.519508	54.162239	995	205	905	705
Fisso 10	104.91088	54.122761	1725	200	-	-
Fisso 11	104.91105	67.927704	1645	100	-	-
Fisso 12	104.88148	43.205231	1800	300	-	-
Fisso 13	99.531624	43.12846	1535	300	275	525
Fisso 14	99.547058	24.85062	1650	520	400	300
Fisso 15	104.88358	24.879362	-	-	130	300
Bianco 16	85.133896	3.8741865	780	925	1070	130
Bianco 17	72.73455	3.8047862	-	-	1570	130
Bianco 18	72.379257	62.353729	320	140	-	-
Bianco 19	85.527481	62.000118	865	145	1110	870
Fisso 20	93.989212	34.106133	1290	400	635	400

Camera Calibration Data, Cont'd

			Camera 3		Camera 4	
	World Coordinates		Image Coordinates		Image Coordinates	
Fisso 1	52.397995	0.0152318	945	1025	960	105
Fisso 2	52.370514	24.917166	945	520	955	300
Fisso 3	54.017929	35.079655	945	400	955	400
Fisso 4	52.352539	42.985855	945	300	950	525
Fisso 5	52.342651	67.981934	945	100	935	1035
Punto 6	61.143108	6.5223298	1590	855	585	145
Punto 7	69.373505	12.066032	-	-	195	185
Punto 8	66.242065	33.832989	1710	400	200	395
Punto 9	69.16835	56.078075	1715	180	-	-
Punto 10	61.844486	60.90424	1360	150	240	855
Punto 11	43.337059	60.931126	550	150	1605	860
Punto 12	34.635906	56.214333	135	185	-	-
Punto 13	38.320267	33.869808	170	400	1720	400
Punto 14	34.079475	11.767772	-	-	1785	180
Punto 15	43.386677	6.6477127	275	855	1350	150

Camera Calibration Data, Cont'd

			Camera 5		Camera 6	
	World Coordinates		Image Coordinates		Image Coordinates	
Giallo 1	74.0178	24.034	1690	800	525	170
Giallo 2	27.8374	34.5678	1525	390	405	405
Giallo 3	28.2874	58.0262	1450	170	195	790
Giallo 4	7.73866	57.9556	565	175	1620	785
Giallo 5	7.58353	9.54519	280	795	1370	175
Fisso 7	-0.01735	13.77	-	-	1720	205
Fisso 8	16.3643	13.7746	895	705	1000	205
Fisso 9	16.3566	54.0618	935	200	1025	705
Fisso 10	0.0786684	54.0799	205	200	-	-
Fisso 11	0.105432	67.9509	290	100	-	-
Fisso 12	0.0919561	43.1069	125	305	-	-
Fisso 13	5.35106	43.0971	395	300	1650	520
Fisso 14	5.35488	24.779	270	525	1535	300
Fisso 15	0.007198	24.751	-	-	1795	305
Bianco 16	19.4079	4.83892	1105	895	875	140
Bianco 17	32.8564	4.60142	-	-	335	135
Bianco 18	32.9241	63.4719	1625	125	-	-
Bianco 19	19.5252	63.5554	1070	130	800	910
Fisso 20	10.837	33.9567	625	400	1300	400

Camera Calibration Data, Cont'd

Focal Length in x = 953.58
Focal Length in y = 953.58
Principal Point = (959.5, 539.5)

	Rotation Matrix			Position (metres) $[x\ y\ z]^T$
Camera 1	0.99997	−0.00769	0.00016	87.56
	−0.00500	−0.66628	−0.74569	−34.52
	0.00585	0.74567	−0.66630	60.69
Camera 2	−0.99998	$7.600e-5$	0.00628	87.69
	−0.00475	0.66371	−0.74797	103.14
	−0.00411	−0.74799	−0.66370	60.62
Camera 3	0.99997	−0.00564	−0.00567	53.15
	−0.00797	−0.64563	−0.76361	−34.13
	0.00065	0.76363	−0.64565	56.51
Camera 4	−0.99990	−0.00507	0.01340	52.37
	−0.01345	0.65367	−0.75666	101.78
	−0.00493	−0.75676	−0.65367	57.93
Camera 5	0.98952	0.00664	0.14427	9.31
	0.11249	−0.66193	−0.74109	−33.97
	0.09058	0.74954	−0.65574	59.50
Camera 6	−0.98373	0.00169	0.17965	27.84
	−0.13250	0.66846	−0.73185	102.09
	−0.12133	−0.74374	−0.65736	58.83