Visually tracking football games based on TV broadcasts

Michael Beetz, Suat Gedikli, Jan Bandouch,
Bernhard Kirchlechner, Nico v. Hoyningen-Huene, Alexander Perzylo
Intelligent Autonomous Systems Group
Technische Universität München, Munich, Germany
aspogamo@in.tum.de

Abstract

This paper describes ASPOGAMO, a visual tracking system that determines the coordinates and trajectories of football players in camera view based on TV broadcasts. To do so, ASPOGAMO solves a complex probabilistic estimation problem that consists of three subproblems that interact in subtle ways: the estimation of the camera direction and zoom factor, the tracking and smoothing of player routes, and the disambiguation of tracked players after occlusions. The paper concentrates on system aspects that make it suitable for operating under unconstrained conditions and in (almost) realtime. We report on results obtained in a public demonstration at RoboCup 2006 where we conducted extensive experiments with real data from live coverage of World Cup 2006 games in Germany.

1 Introduction

As more and more computer systems are equipped with sensing devices and are installed in environments where people act, there is a steadily increasing interest in application systems that can automatically interpret and analyze intentional activities. Examples of such application domains are human living environments where computer systems are supposed to support the people's everyday life, or factories where machines and human workers are supposed to perform the production processes in joint action. Another class of example domains that starts to receive increasing attention is the automated analysis of sport games.

In all these cases the perception of intentional activity has to be solved reliably, accurately, and under unmodified conditions. Indeed scaling reliability and accuracy of such perception towards real life situations still poses challenging problems for most perception and AI systems.

In this paper we consider a particular subclass of these problems, namely the accurate and reliable tracking of players based on broadcasts of football games. To address the problem, we have realized ASPOGAMO (Automated SPOrt Game Analysis MOdel), a computer system estimating motion trajectories in world coordinates for the players in the camera view.

ASPOGAMO can track the players based on views provided by the overview cameras used in game broadcasts. It has a high player detection rate of more than 90%, an average inaccuracy of less than 50 cm, handles many cases of occlusions, and works in almost realtime.[1] ASPOGAMO deals with substantial variations in lighting conditions and complicated team dresses. The system has been demonstrated at the RoboCup 2006 in Bremen, tracking players based on live broadcasts of FIFA World Cup 2006.

The purpose of this paper is to briefly describe the basic system and algorithms and then to discuss in detail the extensions that are made to scale towards a real-world application. The main contributions are threefold:

1. an accurate and reliable model-based estimation system for the camera parameters based on the video sequence,
2. a color-based segmentation system for robust and precise detection of players in camera view even under challenging conditions (difficult lighting and colors, partial occlusions, camera motion blur), and
3. a probabilistic (multi-camera enabled) player tracking system that handles a wide range of game typical occlusion scenarios.

The remainder of this paper is organized as follows. The next section introduces the software architecture and the basic solution idea to solve the tracking problem. Then the underlying components provided for obtaining the necessary reliability and accuracy are detailed in the sections 3 (estimating camera parameters), 4 (detecting players), and 5 (tracking players). We will then describe experimental results, discuss related work and finish with our conclusions.

2 System Overview

Before we discuss the technical details of our approach we will first explain the computational problem that we are solving, the structuring of the problem into subproblems, and the basic mechanisms for solving the subproblems.

2.1 Input and output

The broadcasted video stream mainly consists of stretches recorded by the main camera, which is placed 8-22m high in the stands near the center line of the field. The main camera

[1] A more specific description of the performance aspects and experimental results is given in section 6.

records the game activities and provides a good overview of the game action. The stream, however, is interrupted through displays of close-ups and replays from other cameras, usually when the game is paused or the activity is not interesting.

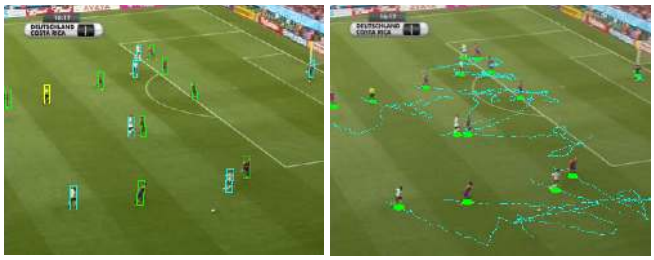

Figure 1: Player detection (left) and created tracks (right)

ASPOGAMO is given image sequences of the game taken by the main camera. The camera's position in the stadium is fixed, but it is constantly pointing towards the main area of activity and changing its zoom factor. Therefore, the pan and tilt angles as well as the focus of the camera are continually changing. ASPOGAMO computes trajectories on the field plane, representing the positions over time of individual players captured on camera (see Figure 1).

2.2 Software Architecture

ASPOGAMO consists of four components: Vision, State Estimation, Track Editor and Motion Model Generator (Figure 3).

The two main components that will be described in this paper are the *Vision* (responsible for image processing tasks) and the *State Estimation* (used to estimate camera parameters and player positions as well as forming player tracks). They interact strongly to simplify each subtask.

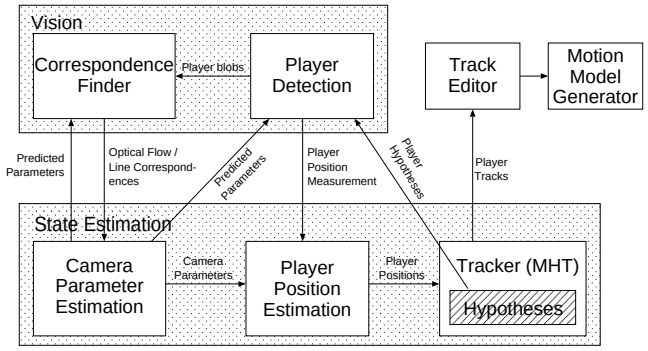

Figure 2: Software architecture of the ASPOGAMO system.

ASPOGAMO processes a video-stream by performing the following steps for each image. First, the *Camera Parameter Estimation* predicts camera parameters for the current frame using optical flow from the *Correspondence Finder*. Based on these predictions, the system uses context information and model assumption in order to simplify the player detection and the search for line correspondences. The *Player Detection* determines player blobs taking into account the hypotheses generated by the *Tracker*. Then, the *Correspondence Finder* uses predicted parameters and the player blobs to find line correspondences considering occlusions. These are used in conjunction with the given prediction based on optical flow to determine the a posteriori camera parameters and their uncertainties in terms of covariances in the *Camera Parameter Estimation*. Knowing the camera parameters, player positions and their uncertainties are calculated in the *Player Position Estimation* from the observations made by the *Player Detection*. They are forwarded to the *Tracker*, where individual positions are combined and associated over time, and hypotheses for the next time step are generated. Once the *Tracker* completes the tracks, they are send to the *Track Editor*, where they are associated with other tracks and receive player IDs in a semi-automatic process. Afterwards, completed tracks are smoothed and stored as a compacted representation in the *Motion Model Generator*.

3 Camera Parameter Estimation

In order to transform image coordinates into real-world coordinates, we need the position and direction of the camera as well as its intrinsic parameters including zoom factor, pixel sizes, principal point and radial distortion coefficient. Because the position of the main camera is fixed during the game, only the direction (pan and tilt) and the zoom factor have to be estimated continuously, as they constantly change while the camera tracks the game activity. All other parameters are determined beforehand in a semi-automatic process.

3.1 Basic Algorithm

ASPOGAMO uses modelbased localization for estimating camera parameters: an image of the game (Figure 3 upper left) and a model of the football field (upper right) are matched together (lower left), and the camera parameters are estimated from this match (lower right).

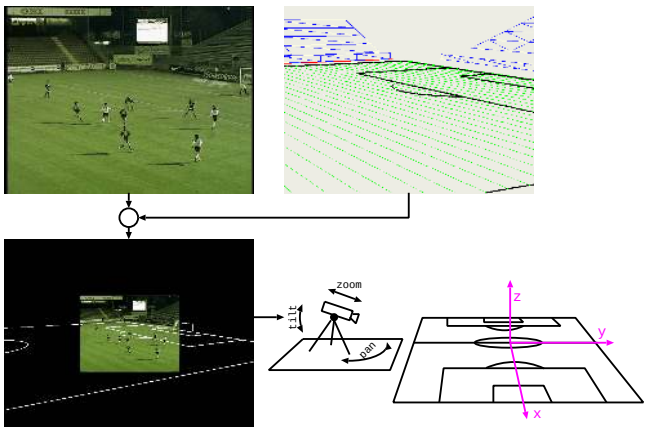

Figure 3: Modelbased localization

The estimation is formulated as an iterative optimization problem. First, the model is projected onto the image using predicted parameter values. The optimizer then searches for image points that correspond to given model points (Figure 4), e.g. on field lines. Finally, the best match between model and image is found by minimizing the distance errors

obtained from the correspondences. This is done using the Iterative Extended Kalman Filter (IEKF). IEKF also gives the uncertainty of the estimation as a covariance matrix, which is used amongst others to determine the search window for correspondences in the next iteration.

Figure 4: Projected fieldmodel using predicted parameters (left), finding correspondences along the searchlines (right).

3.2 Improvements

Figure 5 shows why the basic algorithm for matching the field model to the image lines is unreliable. Non homogeneous lighting conditions, cast shadows and high color variations cause the color classes to be diffuse and to overlap. Segmentation becomes inaccurate or ambiguous, and eventually fails. Low resolution and the inclined camera position causes distant lines to almost disappear in the images. Field lines do not appear as white, but as a slightly lighter green, making them hard to be found by uninformed line detectors. Furthermore, zoomed in camera views often lack visible lines that allow for unambiguous parameter estimation. Motion blur caused by fast camera movements is another problem. Also, the varying focal length of a camera when zooming changes the radial distortion coefficient. Finally, TV broadcasting and video compression produces noisy images with pallid colors.

Figure 5: Some hard scenes: different colors in background (upper left), cast shadow (upper right), motion blur (lower left), not enough field features (lower right)

These difficult context conditions require that

- lines contained in the image must be found reliably, which implies that outliers and false correspondences must be suppressed,
- camera motion must be tracked without the help of model information if not enough field lines are visible,
- the system should be able to recover from failure.

Let us consider some extensions that address these requirements in more detail and result in reliable longterm tracking of the camera.

Reliable line detection: To deal with blurry lines and edges we apply probabilistic methods for line detection. Rather than making a hard color classification we assess the probability that pixels contain field lines. As stated above, distant white lines often appear as lighter green. We model color classes as a mixture of Gaussians in RGB colorspace (see 4.1). ASPOGAMO detects lines by searching points where color distributions on both sides apply to the color green with a higher intensity in between. Matches are thresholded depending on the expected linewidth in pixels. The quality of a correspondence and its variance is estimated depending on the match and its uniqueness. A similar method is used for edge detection.

To reduce the influence of outliers (from a quadratic to a linear one), we're using weights for each correspondence according to the weight function from Huber [6] from the M-Estimators [11]. Assuming distances (errors/costs) of outliers are large, the M-Estimators suppress such correspondences.

Tracking without field lines: For prediction we use the optical flow from two subsequent images. Only 100 randomly spread pixels from around the image border are used. During fast camera movement the rate of outliers increases, and without the outlier suppression the camera would be lost in a few steps. As the outlier suppression proposed in the upper section has no context information and considers only the distance for each correspondence, it might suppress correct correspondences having large distances while keeping erroneous correspondences having small distances. To address this problem we estimate the most likely homography to transform points between the two image planes, which is used for estimating correspondence variances and filtering outliers. This homography can be calculated using only 4 point-correspondences in the image plane. The RANSAC algorithm [5] is a robust method to find out the most likely homography.

Failure recovery: When the inaccuracies in the parameter estimation increase, the quality of the correspondences decreases as the projected field model starts drifting away from its real position. We detect these cases where we lose the tracking of the camera. A semi-automatic process is used to reinitialize the camera parameters. We currently integrate a fully automated field matching process into ASPOGAMO.

4 Player Detection

ASPOGAMO detects players by segmenting blobs on the field that are not field green. Candidate blobs are analyzed using size constraints and color models. Players are localized using their estimated center of gravity.

4.1 Used Color Model

Since football is a spectator sport, the playing field, the lines and the dresses of the players are designed to be visually distinctive. The most informative visual feature herein is color: the field is green, the lines are white, and the teams should dress so that they can be distinguished easily. We exploit the situation by designing algorithms that use color as their primary visual feature.

We model color classes as mixture of Gaussians in RGB space. This gives us the option to use statistics and probabilities in both learning and using color models. Having multimodal distributions is important e.g. when trying to segment a field with cast shadows from a stadium roof (Figure 6). It also helps coping with lighting changes and keeping resembling color classes disjunct. For efficiency reasons we scale the RGB space down to the 16bit R5G6B5 space and use precomputed lookup tables for distances and densities. Color models are learned semi-automatically using k-means clustering on manually marked regions. The ability to refine the learned colors as time progresses is inherent. Fully automated color learning will be available in the near future.

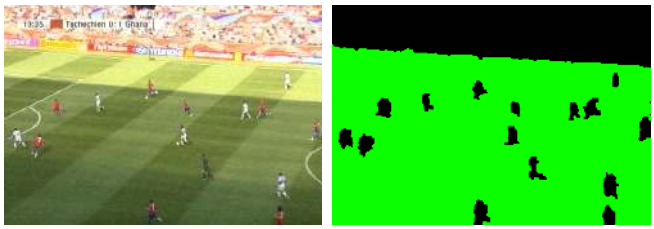

Figure 6: Color segmentation on field with cast shadow

4.2 Recognizing and Localizing Players

As illustrated in Figure 7, simple color-based segmentation of players is doomed to fail. Depending on the camera's position and focus, typical player sizes in the image are around 30 pixels, but can easily be as small as 10 pixels. Input images tend to be very noisy as they are taken directly from TV broadcast. Player's are often occluded by other ones, especially in situations like corners or free kicks. Furthermore, fast camera movement causes additional motion blur. As a result, shapes and colors mix with their neighborhood.

Figure 7: Enlarged players. Pixels are washed-out due to the small resolution and motion blur. Players in front of advertising-banners are even harder to detect (right image).

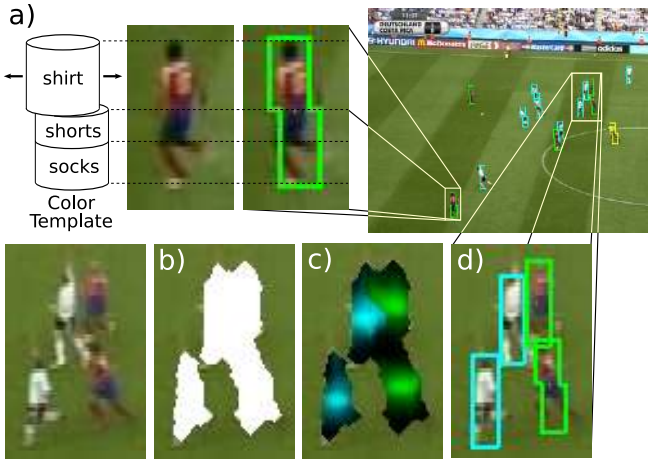

Figure 8: Player recognition.

To solve the player recognition task despite these complications, we combine simple but powerful perceptual cues in a probabilistic manner, in order to achieve robustness, speed as well as the required precision. In a first step, all potential player regions unlikely to belong to either the field or the surroundings are segmented (Figure 8b). We then estimate the mass centers of the players, more precisely the center between shirt and shorts, as this point prove to be the most robust to detect. This is done by calculating a likelihood map for the player whereabouts using multiple evidences based on color template matching, size constraints and predictions (Figure 8c). Player positions are extracted from the map by finding strong modes (Figure 8d) and projecting their coordinates to a point half a player's height over the field plane.

Let us now consider the computation of the likelihood map in more detail (see Figure 9). Potential player regions are found by segmenting the field and intersecting the inverted field region with the field's convex hull. The field region itself is segmented using a hysteresis threshold based on the Mahalanobis distances to the learned field and line colors. Sparse morphology is used to remove noise and to combine field regions still separated by e.g. field lines. The resulting player blobs (Figure 9a) may contain more than one person, none, or only part of a person, when e.g. a leg got cut off because of bad color contrast.

To localize the exact number and positions of players inside the blobs, we use color template matching (Figure 9b), where a template (Figure 8a) for every player type is moved across every pixel in a blob to calculate a similarity measure. The template is scaled to the expected size of a player at the respective image position, which can be calculated from the estimated camera parameters (see Section 3). The player's appearance is approximated by subdividing the template into a shirt, a shorts and a socks region, and associating each of them with the respective color classes learned in chapter 4.1. We use only distinctive color classes, as hair or skin colors do not qualify for separating different player types. The tem-

plate is centered between shirt and shorts, which coincides well with a human's mass center. To account for inclined players, the shirt region of the template is shifted to five different configurations, and the best one is selected. The similarity measure is calculated from all pixels inside the template mask taking into account their probabilities regarding the affiliation to the template's associated color classes.

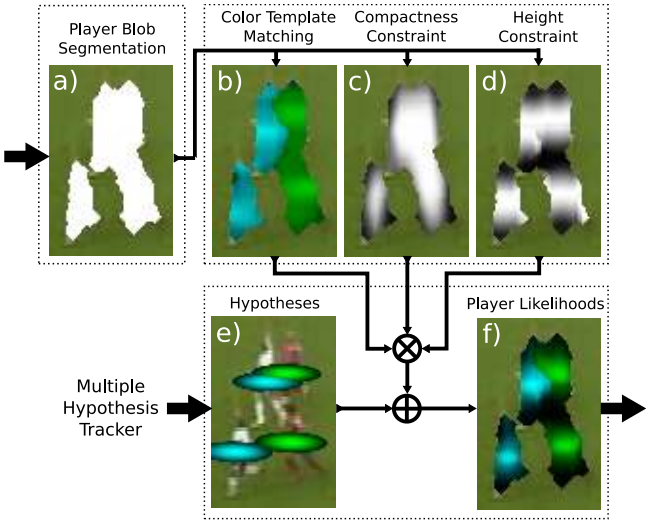

Figure 9: Calculating the likelihood map from Figure 8c. First, player blobs are segmented (a). Then, three likelihood maps based on color template matching (b), compactness (c) and height constraints (d) are calculated. In the last fusion step these are multiplied, and likelihoods at predicted player positions (e) are increased to form the final map (f).

Additional constraints are applied to increase accuracy in situations where players are completely dressed in one color, or players from different teams share similar colors on different parts of their dresses. In such cases, high likelihoods can be observed at positions that do not match the player's locations well. The compactness constraint (Figure 9c) produces low likelihoods wherever the overlap between the player blobs and the template mask is small, reducing high probabilities near the blob contours. The size constraint (Figure 9d) produces high likelihoods at locations where either the distance to the upper blob contour or to the lower blob contour is near the optimum for the expected player size. This is justified by the fact that the heads of the players should be located below the upper contour, and the feet above the lower contour. In case of bad segmentation, e.g. with feet cut off, there is still a good chance that at least the opposing contour has been segmented properly.

Finally, the three likelihood maps are fused by multiplication, and probabilities near the regions where a hypothesis has been returned from our tracker (Figure 9e) are increased. This way, bad observations due to partial occlusion or fast motion can be enhanced. The resulting likelihood map (Figure 9f) is then searched for modes, and if they exceed a predefined threshold, a player observation is forwarded to the tracker. If the regional overlap between two observations is too high, only the stronger observation is selected. The location of the player on the field plane is then calculated using the estimated camera parameters.

There are several issues that deserve discussion: By using color template matching with the additional constraints, players are recognized quite reliably even in the case of slight occlusions. Probabilities and covariances for the observations can be easily extracted from the likelihood map and forwarded to the tracker. False positives and missed detections are rare, and can be dealt with by the tracker. Due to the simplicity of the detection without extensive exception handling, the system generalizes well to different situations, e.g. wide angle or zoomed in camera views. Even cast shadows on the field, motion blur or noisy input data can be dealt with, provided that the colors have been learned accurately.

Although we use template matching on full resolution PAL images (due to the small player sizes), the player detection runs at realtime. The template matching is speed up using integral images thanks to the square template masks. Another contributing performance factor is the use of runlength encoded region masks.

5 Player Tracking

As seen in the previous section, player positions can have big measurement errors, player types can be interchanged, false positive measurements or undetected players can disturb the tracking as well as player clusters. So the tracking of the players is not a straightforward task.

In our system we use a Multiple Hypothesis Tracker (MHT) [8] to account for missing measurements and improve the robustness of the tracking. The input for the MHT are the positions and covariance matrices of the measurements. The output are tags that define the track affiliation for the individual measurements. The smoothing capability of the internal Kalman/particle filter is not propagated because we always generate a motion model afterwards which can produce better approximations to the real path.

The implementation we use is the one described in [2] with some improvements to account for the non-Gaussian distribution in blobs and the ability to assign more than one track to a blob measurement.

6 Empirical Results

We have presented ASPOGAMO at RoboCup 2006, where we gathered large amounts of input data from live coverage of World Cup 2006 games in Germany and conducted extensive experiments under public display.

Camera parameter estimation: The system has been able to track the camera without getting lost in almost all tested scenes from World Cup 2006 (about 30 scenes, each 10 to 40 sec long). Using original TV broadcast camera streams without scene disruptions, the system is able to track the camera parameters up to 20 minutes without getting lost. The accuracy of back-projections on the model-field is about 5 cm to 1.0 m, depending on the camera movement, the visibility of field features and the distance to the camera.

Player detection: Table 1 shows the player detection rates for some games from the World Cup, that have been deter-

mined manually by examining 344 randomly selected frames. Both the opening game Germany vs. Costa Rica and the game Argentina vs. Serbia Montenegro had detection rates over 90%, despite very noisy input data. The game Portugal vs. Iran was challenging because of a strong cast shadow on the field, making white dressed players in the sunlight hard to detect even for the human observer. Finally we examined a corner situation from the opening game. The missed detections are almost entirely associated to occlusions and clustering of players. Our experiments with other games confirm that detection rates of over 90% are generally achievable.

game	player count	missed players	mis-classified	false positives
GER - CRC	2353	4.78%	0.38%	2.37%
ARG - SCG	1605	5.51%	1.88%	1.23%
POR - IRN	592	21.96%	3.72%	0.34%
Corner	462	17.53%	3.68%	6.06%

Table 1: detection rates for players

The presented detection rates are based on observations that do not exploit the temporal information given in our tracker. When incorporating the tracker, most of the errors resulting from temporarily missed detections, false positives or misclassifications are resolved. Figure 1 shows both the detected players and the created tracks for a short scene. In our experiments most of the tracks are complete for as long as a player is in camera view, usually until the scene gets interrupted. Exceptions are players in front of advertising banners or extremely clustered scenes like corners.

The inaccuracy of the player detection is typically lower than 0.5 m. The precision has been estimated by projecting detected image positions and manually determined image positions on the virtual field plane. Another conclusion obtained from the experiments is that if both shirt and short colors are similar, pose estimation in vertical image direction is harder, and the detected positions tend to oscillate more. But the effect is reduced by smoothing and averaging the final tracks in the *Motion Model Generator*.

ASPOGAMO is capable of fusing observations from multiple cameras viewing the same scene from different positions. Experiments have shown that the accuracy of the player detection is further improved, as the uncertainties of the measurements become smaller. ASPOGAMO has also been used to gather all player trajectories over a full game of 90 minutes.

7 Related Work

Several approaches in tracking soccer players exist that are constrained to static cameras [4; 9; 7]. Unfortunately, such systems are complex and expensive in their hardware setup. Farin et al. [3] propose a method for camera calibration in realtime, but without player detection. Bebie and Bieri [1] and Yamada et al. [10] describe similar systems also tracking players from TV broadcast, but both lack experimental results. No other systems comparable to ASPOGAMO in terms of reliability, robustness and performance are known to us as yet.

8 Conclusions

In this paper we have described ASPOGAMO, a probabilistic vision-based tracking system for estimating the positions of players based on broadcasted football games. The main contributions are a set of techniques that make the system so robust that it reliably functions for broadcasts. The empirical studies show that the system can cope with difficult and changing lighting conditions, fast camera motions, and distant players. ASPOGAMO enables computer systems to semi-automatically infer abstract activity models of football games. These models can be used for supporting coaches, football reporters, and interactive television.

References

[1] T. Bebie and H. Bieri. Soccerman - reconstructing soccer games from video sequences. In *ICIP (1)*, pages 898–902, 1998.

[2] I. J. Cox and S. L. Hingorani. An efficient implementation of reid's multiple hypothesis tracking algorithm and its evaluation for the purpose of visual tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(2):138–150, February 1996.

[3] D. Farin, J. Han, and P. With. Fast camera calibration for the analysis of sport sequences. In *Proceedings IEEE International Conference on Multimedia and Expo (ICME)*, pages 482–485, 2005.

[4] P. Figueroa, N. Leite, R. Barros, I. Cohen, and G. Medioni. Tracking soccer players using the graph representation. In *Proceedings of the 17th International Conference on Pattern Recognition (ICPR)*, pages 787–790, 2004.

[5] M. A. Fischler and R. C. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, June 1981.

[6] P. J. Huber. *Robust Statistics*. John Wiley & Sons, N.Y., 1981.

[7] J. Kang, I. Cohen, and G. Medioni. Soccer player tracking across uncalibrated camera streams. In *Proceedings of IEEE CVPR*, 2003.

[8] D. B. Reid. An algorithm for tracking multiple targets. *IEEE Transactions on Automatic Control*, 24(6):843–854, December 1979.

[9] M. Xu, J. Orwell, and G. Jones. Tracking football players with multiple cameras. *International Conference on Image Processing, ICIP 04*, 2004.

[10] 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. In *Proceedings of the 16 th International Conference on Pattern Recognition (ICPR'02)*, volume 1, pages 10303–10306, August 2002.

[11] Z. Zhang. Parameter estimation techniques: a tutorial with respect to conic fitting. *Image and Vision Computing*, 15(1):59–76, January 1997.