2014 IACSIT Paris Conferences

2014 the 4th International Conference on Industrial Technology and Management (ICITM 2014)

2014 the 3rd International Conference on Image, Vision and Computing (ICIVC 2014)

2014 the 6th International Conference on Software Technology and Engineering (ICSTE 2014)

IACSIT
WWW.IACSIT.ORG

A vision framework for the localization of soccer players and ball on the pitch using Handycams

Tiago Vilas[1], J.M.F. Rodrigues[1, 2], P.J.S. Cardoso[1], and Bruno Silva[3]
[1]University of the Algarve, Engineering Institute of Faro, Faro, Portugal
[2]Vision Laboratory, LARSyS, University of the Algarve, Faro, Portugal
[3]Inesting, S.A., Faro, Portugal

ABSTRACT

The current performance requirements in soccer make imperative the use of new technologies for game observation and analysis, such that detailed information about the teams' actions is provided. This paper summarizes a framework to collect the soccer players and ball positions using one or more Full HD Handycams, placed no more than 20cm apart in the stands, as well as how this framework connects to the FootData project. The system was based on four main modules: the detection and delimitation of the soccer pitch, the ball and the players detection and assignment to their teams, the tracking of players and ball and finally the computation of their localization (in meters) in the pitch.

Keywords: Segmentation, Classification, Pattern Recognition, Applications, Soccer.

1. INTRODUCTION

Everyone involved in high performances/competition sports make all the efforts aiming for excellence. Not only the players need to show their skills but also the coach, and the remaining technical staff, need to have their own tools so that they all can perform at higher levels. An on-the-fly match analysis during a game can prove to be a very useful tool, provided that significant information to help the tacking of decision by technical team was extracted. In this sense, the most important teams (the ones with the financial capacity) have systems or staff to acquire that information. On the other hand, for others teams (but not only), an affordable portable system that can (semi-)automatically analyze a game in real time and/or from recorded videos could be of much use.

In the literature it can be found examples that explore some aspects of a match analysis, like video technology for coaching [1], systems that are similar to the one presented in this paper in terms of localization of soccer players and ball [2], including the computerized video analysis of soccer [3], or learning to track and identify players from broadcasted sports videos in other sports, like basketball [4]. For a review on vision-based systems for video analysis and event detection in soccer matches see [5, 6].

This paper describes a small part of the FootData's project [7, 8], which aims to develop a Soccer Resource Planning (SRP) system. The framework presented here aims to compute the localization of the soccer players and ball during a match, using images acquired with at most three Handycams, so that statistical information and automatic feedbacks, about what is happening inside of the pitch, can be generated. One of the main contributions of this work is the integration in a single framework of a system that can track players and ball positions (in meters) using: (a) a single camera that has "*motion*" (pan-tilt) and only can acquire part of the soccer pitch; (b) two (or more) static Handycam that can acquire the entire soccer pitch. The system is integrated with a web application, fully functional in the majority of the most used web browsers. For instance, the web application allows a user, placed anywhere and with a minimal computational system (e.g., a tablet) with a broadband, to correct players assignments that are wrong, e.g., due to camera occlusions.

The remaining paper is divided as follows. Section 2 resumes the FootData project and outlines the web application for the tracking editor. The classification and localization of soccer players and ball are described in Section 3. Conclusions and future work are presented in the final section.

2. FOOTDATA PROJECT AND THE WEB TRACKING EDITOR

Footdata [7, 8] is a project in development by the University of the Algarve, Inesting, S.A. and the soccer coach Domingos Paciência. The goal is to build a web application product for soccer, which integrates two fundamental components of this sport's world: i) a social network (FootData-Social), which provides the typical social network features adapted to the soccer reality, and ii) the professional component (FootData-PRO), which can be considered as a Soccer Resource Planning (SRP), featuring a system for acquisition and processing of information to meet all the soccer management needs. The latter includes (between other things) an automated platform to gather information from games. This platform will be based on a system that will process live images acquired on-site, using a single or a group of cameras placed together (no more than 20cm apart) in the stands, typically Full HD Handycams. One of the Footdata's objectives is to add features which will allow the analysis of the soccer match structure allowing, for example, to take actions to rationalize and optimize the teams' actions regarding the occupation of spaces. However, the main goal is to automatically collect information and on-the-fly alert the technical staff about specific events. The project specifications state that all the above should be presented in a web (browser) environment, accessible from a personal computer or a mobile device (smartphone or tablet). It is important to mention that in the case of the Footdata's professional component there are a few commercial systems that partially integrate some of the project features, e.g., Kizanaro [9] or InStat [10].

The goal of the tracking framework within this project is to acquire the position (coordinates from a reference point) in meters of the players and ball in the pitch, to later generate statistics and other feedbacks. By the project constrains, it was limited to use a maximum of 3 Handycams (preferentially 2) to acquire the data from a single place (usually in a cabin) in the stands. The number of cameras configuration (1, 2 or 3) is made in a web interface (*Traking Editor*), and allows the acquisition of data in different forms: (a) 1 camera - allowing pan and tilt movements of the camera, so it can follow the ball and other important actions. In this case, it is necessary to automatically delimit the visible part of the pitch (and consequently the stands) since, in general, it is not possible to capture the entire pitch, and also due to the movements of the camera. (b) 2 cameras (similar with 3 cameras) – in this case it is possible to capture the entire pitch by concatenating the images from those cameras. Since the cameras are static during the entire game (pan, tilt or zoom actions are not allowed), the user is conducted to select at the beginning of the game, in the web interface, a set of specific keypoints that corresponds to the limits of the pitch (for each camera). Of course, in this case it is not necessary to automatically detect where the pitch and stands were, once this was already done implicitly by the manually selection of the keypoints. It must be mentioned that at this point, assuming that each camera captures at least a midfield, the user only needs to adjust the corner and middle point of the field (4 keypoints for each frame/camera), since the software computes all the remaining points from line intersections and returns the information to the user, for any readjustment if necessary.

Figure 1 top shows one screenshot of the web interface (the implementation is out the focus of this paper; a prototype version can be found with all the technology necessary for the implementation in [7]) where the described process can be done, as well as the correction of the number or team of any player that is wrongly assigned. This interface also performs complementary functions between the players and ball tracking module done in C++ with the OpenCV library [11] and the others FootData modules (see [7, 8]).

3. TRACKING OF SOCCER PLAYERS AND BALL

The tracking of the soccer players and ball is done in four stages: (1) the detection of the soccer pitch, (2) the ball detection and the player's detection and assignment to their teams, (3) the tracking of players and ball and (4) the computation of their localization/coordinates (in meters) in the pitch.

3.1 Soccer pitch detection

Let $I_{RGB}(t)$ and $I_{HSV}(t)$ represent a frame with dimensions $M \times N$, acquired by a single camera (Fig. 1, top row) or the frame resulting from the concatenation of the frames acquired by two (or more) cameras in the stands (Fig. 1, 2nd row, left), where M is the width and N is the height of the image, t is the instant when the frame was acquired, and RGB and HSV the color spaces [12].

The first step was the soccer pitch detection, which consists in (a) the segmentation of the green (grass) regions for each frame using I_{HSV}. This was done by an automatic process, comprising the following steps: (a.1) 100 pitch patches, with a size of $0.05M \times 0.05N$ (with minimum size of 5×5 pixels), were evenly distributed in I_{HSV}. Those patches were used to compute the HSV thresholds intervals that corresponds to the grass as follows. (a.2) For each patch the variance was computed, and the patches with variance higher than 2 were discarded. After this, (a.3) using the remaining patches and discarding outliers, being considered outliers all the patches that have an average bigger than the average of the patches plus the biggest variance between those patches, it was computed the maximum and minimum values for each component of H, S and V. Those maximum and minimum values were then used in a process to define which pixels should be considered as grass, returning (a.4) a binary image I_c, with $I_c = 1$ corresponding to the green (expected grass) pixels.

The next step had the purpose of eliminating the areas outside the pitch. This was carried out in several stages: first (b) a central point of the I_c image was chosen as a seed (please note that in the majority of the frames, the central area of the image corresponds to some zone of the pitch, the exceptions are zooms to players or to the stands). The central point of I_c had to have grass (green color; see above), and recursively from this initial point, using 3×3 neighborhood, all central pixels that had all neighbors green were kept, setting $I'_c = 1$, and the remaining were set to 0 ($I'_c = 0$). See Fig. 2 (in black the green pixels) top row, left (single camera) and right (two cameras) for the corresponding frames shown in Fig. 1.

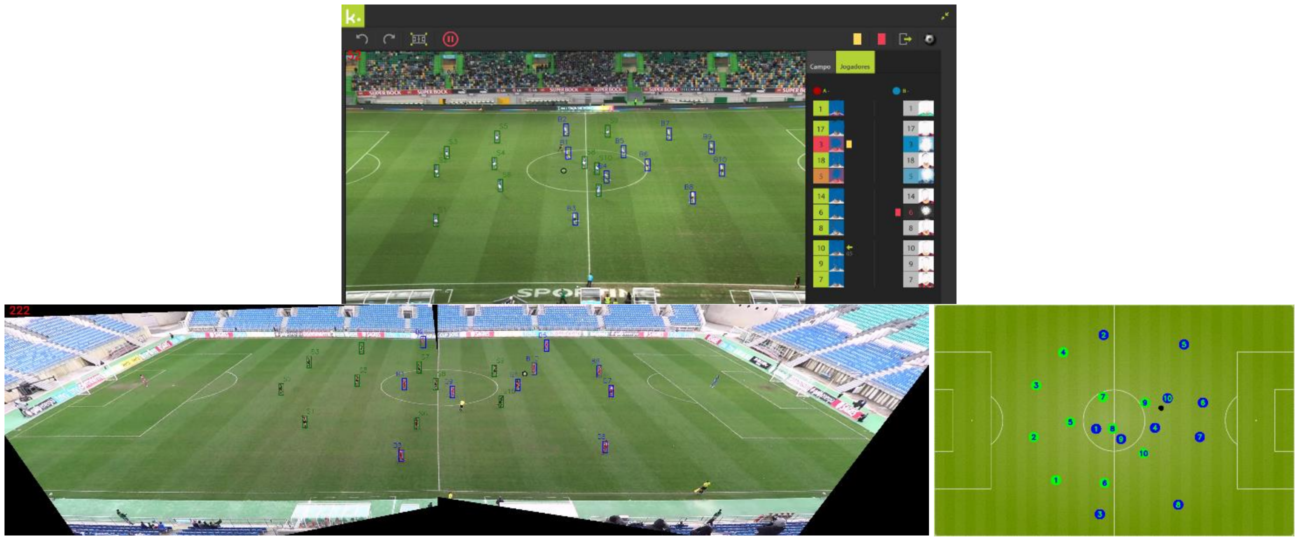

Figure 1. Top, web interface for the *Tracking Editor* with an image showing the results from a single camera. Bottom left, results of the ball and players detection and assignment to their respective team (two cameras). Bottom right, the representation of the real player's position in the pitch for the image on the left.

At this point, two situation can follow. Frames built from more than 1 camera, see step (c.2), or frames from a single camera (c.1). Considering the last case (c.1), (c.1.1) a Canny edge (I_{Ce}) detection [12] was applied on I_{RGB}, where $I'_c = 1$ (e.g., where exists pitch). After this, (c.1.2) the Hough transform [12] was applied over the I_{Ce}, in order to detect the lines (I_{Hl}) that limit the pitch. Finally (c.1.3), I_{Hl} was used to compute the most horizontal and vertical lines. Those lines correspond to the limits of the pitch, and everything outside them was removed. The final results consists in the segmentation of the pitch (with some white blobs inside, corresponding to players), I_{flj}. Fig. 2, 2nd row left, shows the field limit lines and central line (used a reference by the single case camera) marked in gray. Otherwise, if we had the frames from the concatenation of two (or more) static cameras (c.2) and the user manually inserted the pitch limits keypoints (via web interface), then (c.2.1) a perspective transform to a default "real" field (with 105×68 meters) was done to I, providing straight away the segregation of the pitch, I_{flj}, with the some white blobs inside, corresponding to players and/or lines; see Fig. 2, 2nd row right. (d) Now, the known lines/circles/etc. were removed/inhibited from the I_{flj}, returning an image with only the possible players (blobs), I_{fj}.

Two aspects must be mentioned: (i) before the user's manual selection of the keypoints in the two (or three) cameras configuration set, the procedure mentioned in (c.1) is applied to each of the cameras used in the (c.2) case, to computationally estimate the positions of those keypoints. The keypoints estimations are then proposed to the user which has to adjust them, if necessary. (ii) The dimensions of the field limits sometimes change from field to field. Equal in all fields are, e.g., the penalty area, the distance of the penalty spot to the goal or the center circle radius. Those standard dimensions allow us to adapt the above steps to the field size variations, by using at least one of those constant sized lines and/or circles detected to compute an estimation for the correct dimensions (centimeters) that worth a pixel in the frame.

3.2 Players and ball detection and assignment

The I_{fj} image was used for the *players detection*. In this image, most of the work necessary to detect the players was already done since, once the pitch area was delimited, the white blobs inside it are likely players (Fig. 2, 2nd row). Nevertheless, 3 major problems can arise: (i) blobs that are not players, e.g., areas with short grass, (ii) players that "disappear" due to low image definition or being partially equipped in green, and (iii) the overlap of various players.

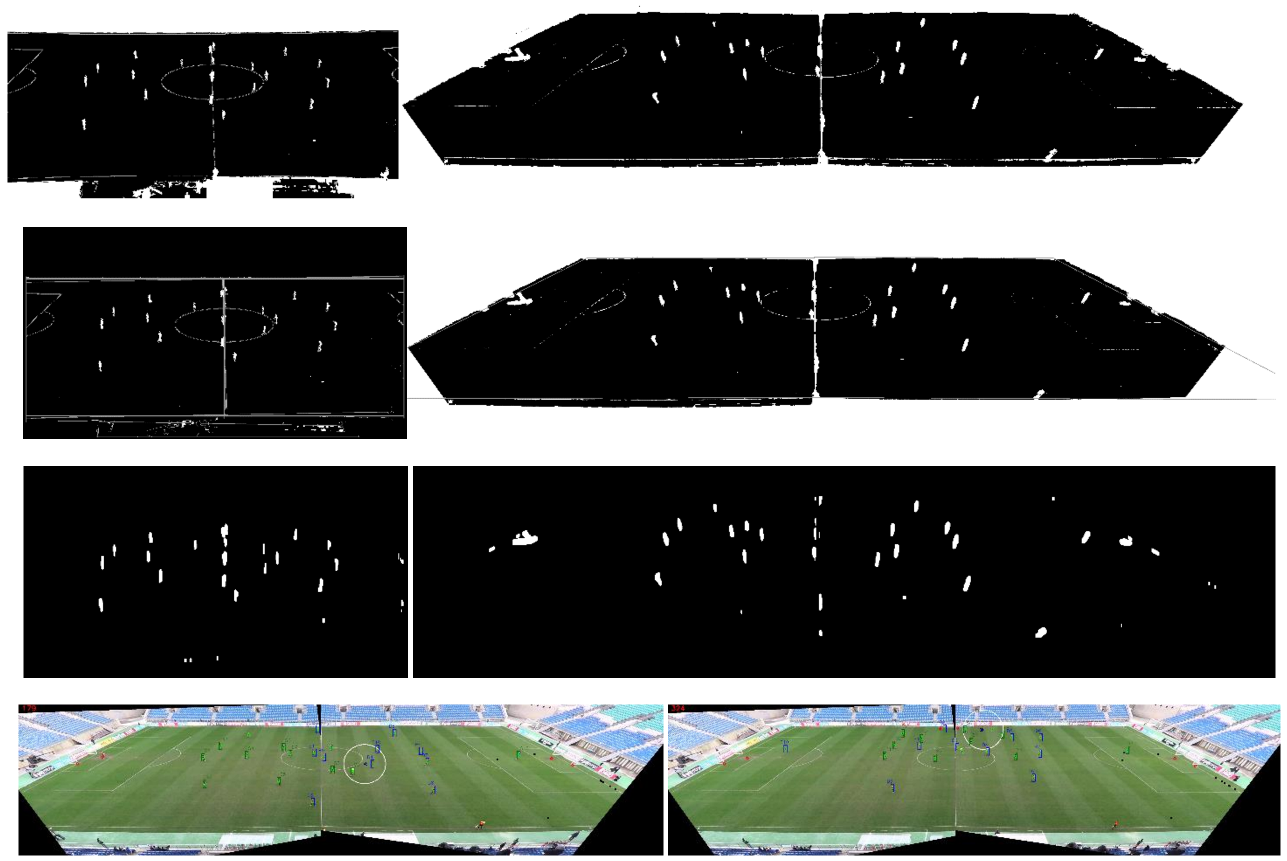

Figure 2. Illustration of some of the most important algorithm steps. The first three rows show images $\boldsymbol{I'_C}, \boldsymbol{I_{flj}}$ and $\boldsymbol{I_{fjmv}}$ (from top to bottom, single camera on the left and two cameras on the right). Last row shows examples of correct player ball assignment with false positives and negatives (see conclusions).

To solve those situations and to obtain only the contours of the players, a set of (a) morphological filters were applied, i.e., first it was applied the dilation filter (D) followed by an erosion filter (E) and finally a 9 × 9 average filter (M). The result was an image, $I_{fjm} = M(E(D(I_{fj})))$, where the players were better defined and specific noise regions were removed. After this, (b) since some player still had more than one region (usually due to the equipment that they use) a vertical mask with the size depending of the position on the field was applied to connect those regions, returning I_{fjmv} (see Fig. 2, 3rd row). The next step was the confirmation that each blob in I_{fjmv} corresponded to a possible player. This confirmation was done applying four conditions, namely: (c) Each blob had at least one edge contour correspondence in I_{Ce}, (d) the height of the blob had to be greater than the weight, (e) the area of each blob had to be between 1/3 and the triple of the average of the detected blobs areas. Finally, (f) it was checked if the blob region (in I_{fj}) doesn't had more than 10% of green pixels (using I_H). The final image after this validation was called I_f. The presented values/parameters were empirically obtained.

In parallel, it was performed the *ball detection* (only predominant white balls, for now). To do the ball detection, all pixels with the "white color" (in I_V), computed only in the field delimited area I'_c, were detected. For this purpose (a) a threshold with V ∈ [75%, 100%] was applied to I_V, returning a binary image, I_b, with the "white regions" inside the pitch identified ($I_b = 1$). (b) The pixels representing lines or circles which belong to the pitch, were removed/inhibited like in step (d) of section 3.1), returning I_{bl}. One downside of doing this line removal was the exclusion of the blob corresponding to ball whenever it was located over a line, which wasn't problematic, once the ball was detected in the flowing frame(s). (c) Also removed were the blobs that corresponded to players (I_f). The next step (d) computed an interval for the ball size in pixels, function of the previously estimated correspondence between a pixel and its value in centimeters in the pitch (see the section above) followed by (e) a test on the ratio/shape of the blobs. The blob that corresponded to the ball had a more or less circular/oval shape (usually!), which was different from the player blobs that had a more "vertical-rectangular" shape. At this point it is important to refer that by using the last criterions, (d) and (e), sometimes the ball was lost, especially when the ball achieved high velocities in broad trajectories. Nevertheless, this only occurred in small sets of frames. Using broader criteria for the thresholds used to detect the ball corresponded to less times lost, but had the drawback of producing more false positives.

Using the described procedures, the majority of the ball and players blobs were detected in each frame. Nevertheless, in some frames more than one ball was detected on the field, situation treated using a tracking algorithm to select the correct ball/blob (see Section 3.4). Figure 1 shows the players delimited by rectangles, and the ball by a circle, for the single camera case (top row) and two cameras frame concatenation (2nd row left). The same representation for Fig. 2 bottom row.

The following step was the assignment of the blobs to the players (number and team), the referee and the keepers. Let N be the number of players' blobs detected in image I_f and B_{p_i} one of those blobs, with i belonging to $\{1, 2, \ldots, N_p\}$, with N_p the number of player. The players' classification to a team was based on the attributes of the blob (B_{p_i}). In more details, for each player's blob, B_{p_i}, I_{VH} was used to compute the following attributes: average value of the (a) Hue and (b) Value for the all blob; the average value of (c) Hue and (d) Value for the superior half of the blob; and the average value of (e) Hue and (f) Value for the inferior half of the blob. For each blob was also computed an array with the length equal to height of the blob, where each entry was the average value of the Value (I_V) in each line of the blob. This array was then smoothed by a moving average period of 4. From this final array, the number of local (g) maxima and (h) minima and their (i) relative position of the each extrema in the array/blob were computed. Finally (also inside of each blob), a (j) single center class for the blob, (k) the single center class for the superior half of the blob, and (l) single center class for the inferior half of the blob were also computed in I_{RGB} using the *k*-means cluster algorithm. Those center classes were then converted to HSV.

Attributes (a)-(l) were then sorted at the beginning of each video/stream of a game, and a subset of attributes ($SA_{\{A,B\}}$) that best fitted a group of 8 blobs and at the same time were the ones that were the most different from those that fitted other 8 blobs group were selected. This two pairs of subsets parameters (values) classified the blobs into the two teams. It was consecutively decreased the test to 7 and 6 blobs if it couldn't find this subsets using 8 blobs. Each blob had to obey at least 90% of the values of the subset attributes that define each team to be assigned to it.

This process of computing the blob attributes, and respective thresholds to select the team was done only once at the beginning of each video/stream part. While in the used benchmark of videos/games there were no major changes in climate, lights, etc., and therefore no update of the parameter was shown necessary, it is expected that this attributes need to be updated periodically during the stream/video. In that case, each attribute measure will be logged and its updates will be weighted using those historical values. Increasing number of updates will hopefully result in less errors in the assignment of the players to their teams, but it has to be weighted since it is a process that consumes CPU.

The keepers were relatively easy to classify and discard once they usually correspond to the first blobs that appear in the right and in the left of the image. Furthermore, since the keepers use equipment different from the teams and referee(s), their blob values were incompatible with the thresholds values computed for the teams. Using the previous process, discarding the referee(s) was also of no major problem. If the video/stream started from the beginning of the game, the referee usually was very close to the middle field line. In those cases, the referee was considered to be the blob with the most different attributes (a-l) from the players near the middle line. If the video started from a different position then the referee was considered to be the blob with the most different attributes (comparing with the attributes from the teams and keepers). If by mistake the referee was assigned as a team player, then the web tool allowed the user to manually change the player classified as referee and vice versa.

The numbers assigned to the players were made in function of the relative position that they occupy in the pitch. Those numbers were associated to the real player number and name in function of the position posted in the game sheet (e.g. right-back). This information is available in the web *Tracking Editor* tool, which communicates with the players and ball detection and assignment module (as mentioned in section 2). If the player had the wrong name or number assigned, this can be corrected later using the already mentioned web tool. Finally, obviously there was also the validation that it was impossible to assign more than 1 field referee, 2 keepers and 10 players per team. Figure 1 shows the result of the player's assignment to a team, where the numbers and players where put from 1-10, and the players faces and teams intentionally presented incorrect due to the project confidentiality agreement.

3.3 Players' positions in the pitch

In order to make a correct analysis of a soccer game it is important to know, as accurate as possible, the correct position of the players in the pitch. That position was calculated using a perspective transformation (homography) [12] from each frame to a normalized pitch. For this purpose a set of references in the pitch was computed. Those references were extracted from the lines (and circles) detected with the Hough transform, I_{Hl}, for the pitch delimitations in I_f in the case of images from a single camera, or from the manual keypoints in the case of images from multiple cameras (see section 2). For the players and ball coordinates presented in the Fig. 1, it was considered the referential origin at top-left corner of the pitch, and the middle point of the bottom line of the box that limits the players for each player, which corresponds in most cases to the coordinates of the players' feet.

Since the pitches have different sizes (weight × height), it was important to compute the size of a pixel. This computation was done in the first frames (as soon as possible) using the lines of the centre circle, penalty area or goal areas which had to be automatically detected to calibrate the value of a pixel in meters, using the fact that their dimensions were the same for all pitches. An automatic process was then applied every time there was a zoom. Figure 1 bottom left, shows the position of the players in a normalized field for the players shown in the right.

3.4 Tracking of players and ball

The players and ball assignment didn't require to be repeated for all frames. Though simple, the implemented tracking process was now effective, except in situation of great confluence of player, such as corners or free kicks. Nevertheless, the main focus of this framework is not to follow the players everywhere, but to compute the distances between players and from the players to the ball during open plays. The tracking was divided into five steps for each frame *t*: (a) The distance (in meters) was calculated between each player in the previous frame, *t* - 1, to the players in the current frame, *t*. Using those distances, each player was tracked by making it correspond to the nearest player in the previous frames. If there were two or more players at the same distance (non-overlapping) the player was assigned to the one that, in the previous frame, had the nearest subset attributes (SA). (b) If there was no players in frame *t* near to the position occupied by some player in frame *t*-1 (few meters apart), then a player with the same color as the one in frame *t*-1 was looked up in frame *t*, using circular areas with increasing diameters and center at the original coordinates. Furthermore, it was used a trajectory computed from the previous 5 frames to limit within the circle the "sectors" of the search. (c) For situations in which a particular player was not tracked/located during a long frame set (2 seconds), this player was then searched from the position where he was "lost" using again circular areas with increasing diameters. The first blob found that was not tracked and had the same SA value was assigned as the lost player. (d) It is important to note that in the majority of the frames for the single case camera, there was the possibility that some players were not detected because they are not in the field of view of the camera, due to fact that the camera doesn't capture the entire pitch. Every player that appear at the left or right (top or bottom, if there was a zoom applied) was considered as new player to track, except if a player was "lost" on a previous frame in a nearby position. This step it is not considered for the 2 (or more) cameras where all the pitch is acquired. (e) Finally, it was also necessary to verify the existence of collisions. If after the above procedure (a-c), a player was not found, it was assumed that there was a collision (junction of two or more players). For those situations, all blobs that correspond to players, that were in positions very close and were lost, were associated with a single blob, assigning to two or more players that were tracked in *t*-1 in a nearby position. When the collision ends (separation of two or more players) the attributes of each blob were checked and players reassigned.

After a collision there was always the possibility of players being wrongly identified, for instance when the collision was between two players of the same team. To solve this, the already mentioned *Traking Editor* web tool (see Fig. 1 top) can be used to correct the players assignment. The process of tracking the ball and referee is similar to the players.

4. CONCLUSION

This paper presents a proposal for detection and tracking of soccer players and ball with the objective of computing distances between players and between the players and the ball. The system was tested to extract game statistics, and it proved to have enough accuracy for that objective. But for now, it stills a framework needing improvements, especially in situations where a confluence of players to a spot of the pitch occurs, such as corners kicks and discussions with the referee. Nevertheless some tests were carried out showing promising results. Depending on the tests done, 1 or 2 cameras, depending also of the team equipment, the results change, nevertheless the detection of players was around 85% of correctness, with around 2.0% of incorrect team assignments. Figure 2 bottom row shows two examples of detections and corresponding assignments (along with some false positives and negatives), where the red circles represent regions detected, but discarded, as players, the green circle represent regions detected and assigned as players, the black squares represent regions detected as possibly being the ball, but not assigned as it, the blue square are regions detected and assigned as being the ball, the green and blue rectangles are detected players assigned to a team (green/blue team), and the game ball is marked by a black circle. Finally, the white circle is the region where in the following frame the ball is going to be searched.

All the steps presented can be subject to improvements, but for now the fundamental focus of improvement is the "Tracking of players and ball" algorithm (Section 3.4).

ACKNOWLEDGMENTS

This work was partly supported by the Portuguese Foundation for Science and Technology (FCT), project LARSyS PEst-OE/EEI/LA0009 /2013 and project FootData QREN I\&DT, n. º23119. Thanks also to project leader Inesting, S.A. [www.inesting.com], and the consultant soccer coach Domingos Paciência.

REFERENCES

[1] B. Wilson, "Development in video technology for coaching," Sports Technology, 1(1), pp. 34-40 (2008)

[2] M. Beetz, N. von Hoyningen-Huene, B. Kirchlechner, S. Gedikli, F. Siles, M. Durus and M. Lames, "Aspogamo: Automated sports game analysis models," Int. J. of Computer Science in Sport, 8(1), pp. 1-21 (2009)

[3] D. Duh, S. Chang, S. Chen and C. Kan, "Automatic broadcast soccer video analysis, player detection, and tracking based on color histogram," In Intelligent Technologies and Engineering Systems, Springer LNCS, vol. 234, pp. 123-130 (2013)

[4] W. L. Lu, J. A. Ting, J. J. Little and K. P. Murphy, "Learning to track and identify players from broadcast sports videos," IEEE Tr. PAMI, 35(7), 1704-1716 (2013)

[5] S.F.S. Júnior, A. Arajo, and D. Menotti, "An overview of automatic event detection in soccer matches," IEEE Workshop on Appl. of Computer Vision, pp.31-38 (2011)

[6] T. D'Orazio and M. Leo, "A review of vision-based systems for soccer video analysis," Pattern recognition, 43(8), pp. 2911-2926 (2010)

[7] P. Rodrigues, A. Belguinha, C. Gomes, P. Cardoso, T. Vilas, R. Mestre, J.M.F. Rodrigues, "Open Source Technologies Involved in Constructing a Web-Based Soccer Information System," In Advances in Information Systems and Technologies SE - 66, vol. 206, pp. 715-723. Springer Berlin Heidelberg (2013)

[8] P. Rodrigues, P. Cardoso and J.M.F. Rodrigues, "A Field, Tracking and Video Editor Tool for a Soccer Resource Planner," In: 8th Iberian Conf. on Information Systems and Technologies, Lisbon, Portugal, 19-22 June, 1, pp. 734-739 (2013)

[9] Kizanaro, http://www.kizanaro.com (2014)

[10] InStat, http://www.instatsoccer.com/ (2014)

[11] OpenCV, http://opencv.org/(2014)

[12] N. Sebe and M. Lew, "Robust computer vision: Theory and applications," The Netherlands: Kluwer Academic Publishers (2003)