Cooperative Detection and Identification of Obstacles in a Robotic Soccer Team

João Silva
Institute of Electronics and Telematics Engineering of Aveiro
Department of Electronics, Telecommunications and Informatics
University of Aveiro, Portugal

Nuno Lau
Institute of Electronics and Telematics Engineering of Aveiro
Department of Electronics, Telecommunications and Informatics
University of Aveiro, Portugal

António J. R. Neves
Institute of Electronics and Telematics Engineering of Aveiro
Department of Electronics, Telecommunications and Informatics
University of Aveiro, Portugal

1 Introduction

Within the many existing research domains in the area of multi robot systems, robotic soccer is one of the most popular ones. One of the existing initiatives within the area is the RoboCup. The RoboCup[1] is an international joint project to promote artificial intelligence, robotics and related fields. Most of the RoboCup leagues have soccer as platform for developing technology, either at software or hardware levels, with single or multiple agents, cooperative or competitive (Kitano et al., 1997).

Among RoboCup leagues, the Middle Size League (MSL) is one of the most challenging, due to its characteristics in terms of rules and environment. In this league, each team is composed of up to 5 robots with maximum size of 50x50cm base, 80cm height and a maximum weight of 40Kg, playing in a field of 18x12m. The rules of the game are similar to the official FIFA rules, with required changes to adapt for the playing robots (MSL Technical Committee, 2010).

Each robot is autonomous and has its own sensorial means. They can communicate among them, and with an external computer acting as a coach, through a wireless network. This coach computer, however, cannot have any sensor, it only knows what is reported by the playing robots. The agents should be able to evaluate the state of the world and make decisions suitable to fulfil the cooperative team objective.

The work described in this document was accomplished within the MSL context. CAMBADA, *Cooperative Autonomous Mobile roBots with Advanced Distributed Architecture*, is the MSL Robotic Soccer team from the University of Aveiro. The project started in 2003, coordinated by the IEETA[2] ATRI[3] group and involves people working on several areas for building the mechanical structure of the robot, its hardware architecture and controllers and the software development in areas such as image analysis and processing, sensor and information fusion, reasoning and control.

Being soccer a very dynamic scenario, our robots need to be able to perceive the other robots around them, both opponents and the own team mates and need to move effectively around the field, meaning they must move (either freely for repositioning or dribling the ball) while avoiding contact with any other obstacle on the field, either robot or even the referee.

For the CAMBADA team to improve its performance during the games, we felt the necessity to improve the detection and sharing of obstacles among team mates. Since we make use of team formations, which keep the robots strategically spread around the field, we wish to ensure a global idea of the field occupancy for all the team. If we achieve a good cover of the field and pinpoint the obstacles inside it, passlines and dribbling corridors can be estimated more easily allowing improvements on team strategy and coordination.

This is essentially an information fusion problem, as the information available from the sensors of each robot and the shared information must be matched and refined. The final result of this fusion is a representation of the state of the surrounding world. In the CAMBADA team, there is an integration process responsible for that task. It is a step executed after image analysis and is responsible to take raw information from the vision and other robot sensors and make a sensor fusion of all sources. For that, it may use the values stored in the previous representation, the current sensor measures (possibly after pre-processing) that has just arrived, the current actuator commands and also information that is available from other robots sensors or world state.

[1] http://www.robocup.org/

[2] Instituto de Engenharia Electrónica e Telemática de Aveiro - Aveiro's Institute of Electronic and Telematic Engineering

[3] Actividade Transversal em Robótica Inteligente - Transverse Activity on Intelligent Robotics

Figure 1: Picture of the team robots used to obtain the results presented on this chapter.

All the information available from the sensors in the current cycle is kept in specific data structures (Figure 2), for posterior fusion and integration, based on both the current information and the previous state of the world.

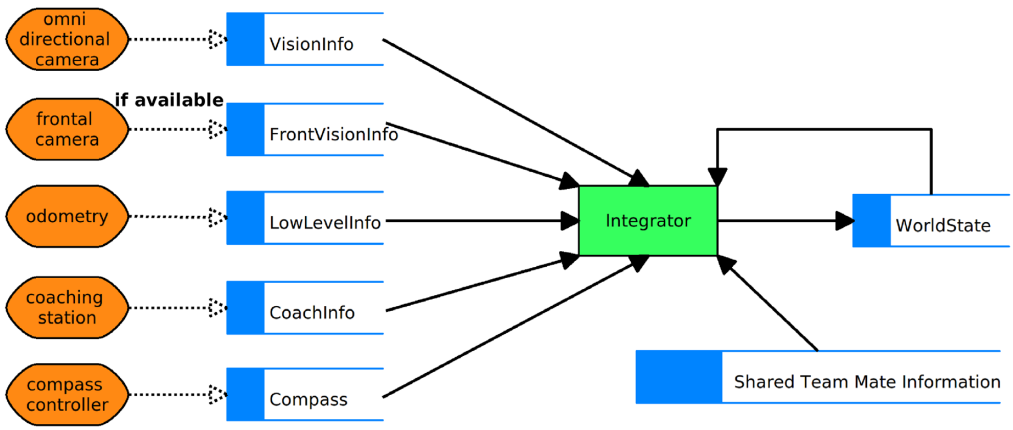


Figure 2: Integrator functionality diagram.

This chapter focuses on the description of the obstacle treatment in the CAMBADA team. In Section 2 a general description of how the obstacles are detected by the vision process is presented, also describing how the raw information is passed to the integration process. Section 3 describes how the raw information is read and, based only on the local robot information, how the identification is processed. Section 5 presents how the sharing and acceptance of team mates obstacles is made. Section 6 concludes this document.

Note that most of the Figures presented in this chapter are images acquired by the omni-directional camera of the robots. To ensure the comprehension of the Figures, in most of them, lines were made around the areas of interest of the image. Also, the triangles and squares representing the obstacle centres and limits inside that areas of interest were increased in size and intensity, since the original image capture squares are more difficult to see in images of the presented size.

2 Visual Obstacle Detection

In the CAMBADA team, visual information about the obstacles is gathered by the omnidirectional camera. According to RoboCup rules, the robots are mainly black. Since in a game robots play autonomously, all obstacles in the field are the robots themselves (and occasionally the referee, which is recommended to have black/dark pants). The CAMBADA vision algorithm takes advantage of this fact and detects the obstacles by evaluating blobs of black color inside the field of play (Neves et al., 2007).

The algorithm for detecting objects in the CAMBADA omni vision system is based in color segmentation. Several items of interest are detected based on their color: the green field, the white lines, the black robots and the defined colored ball. From the center of the image, which corresponds to the center of the robot, radial sensors are created around the robot, each sensor represented by a line with a given angle (Figure 3). These lines are called *scanlines* (Neves et al., 2008) and the color of the pixels on them is analyzed, in search for the defined colors or an undefined one. In the obstacles case, the lines are searched for black color.

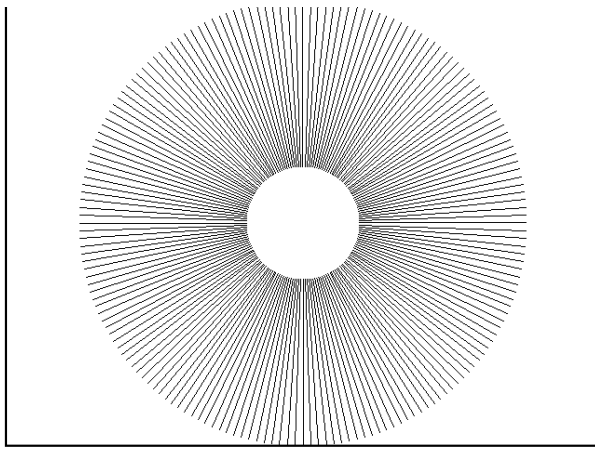


Figure 3: Picture representing an instance of *scanlines*, which work as a mask overlapped with the segmented camera image. Only the pixels on that mask will be analyzed.

2.1 Obstacles detection using only visual information

The detection of black color on the scanlines was analyzed both in angular intervals and length intervals, to define the limits of each black blob. Since the size of the objects on the image vary with the distance

to the robot, it is advantageous to know the relation of that variation. The omni vision system is a non-SVP hyperbolic catadioptric system. Through an inverse distance map calculation, by exploring a back-propagation ray-tracing approach and the physical properties of the mirror surface (Cunha et al., 2008), the relation between the distances in the image and the distances in the real world is known (Figure 4).

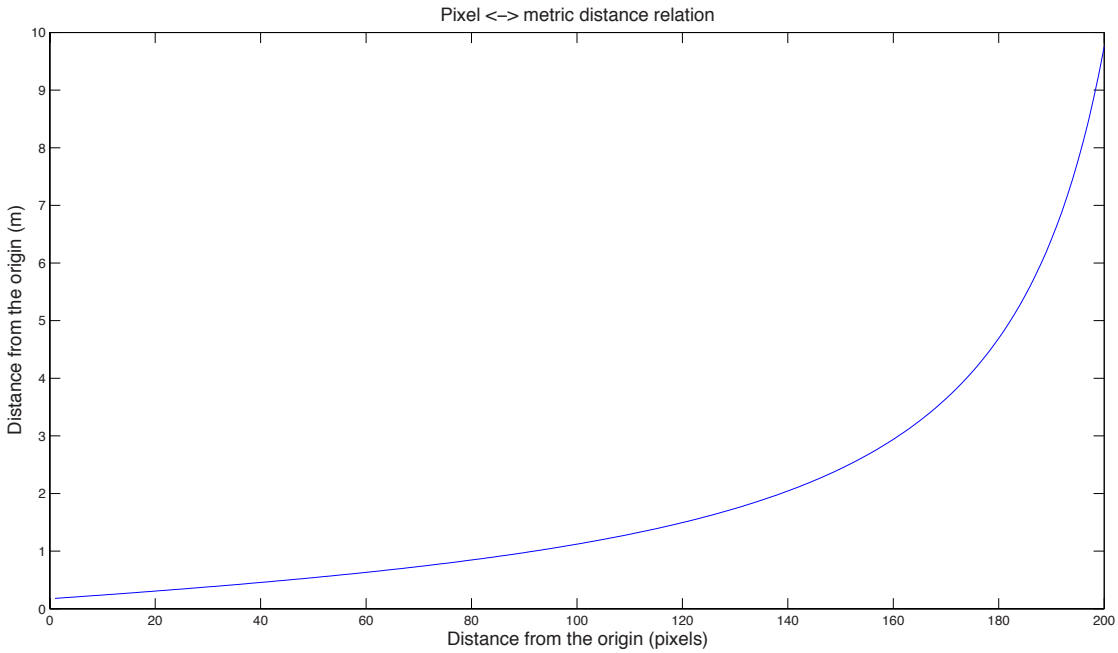


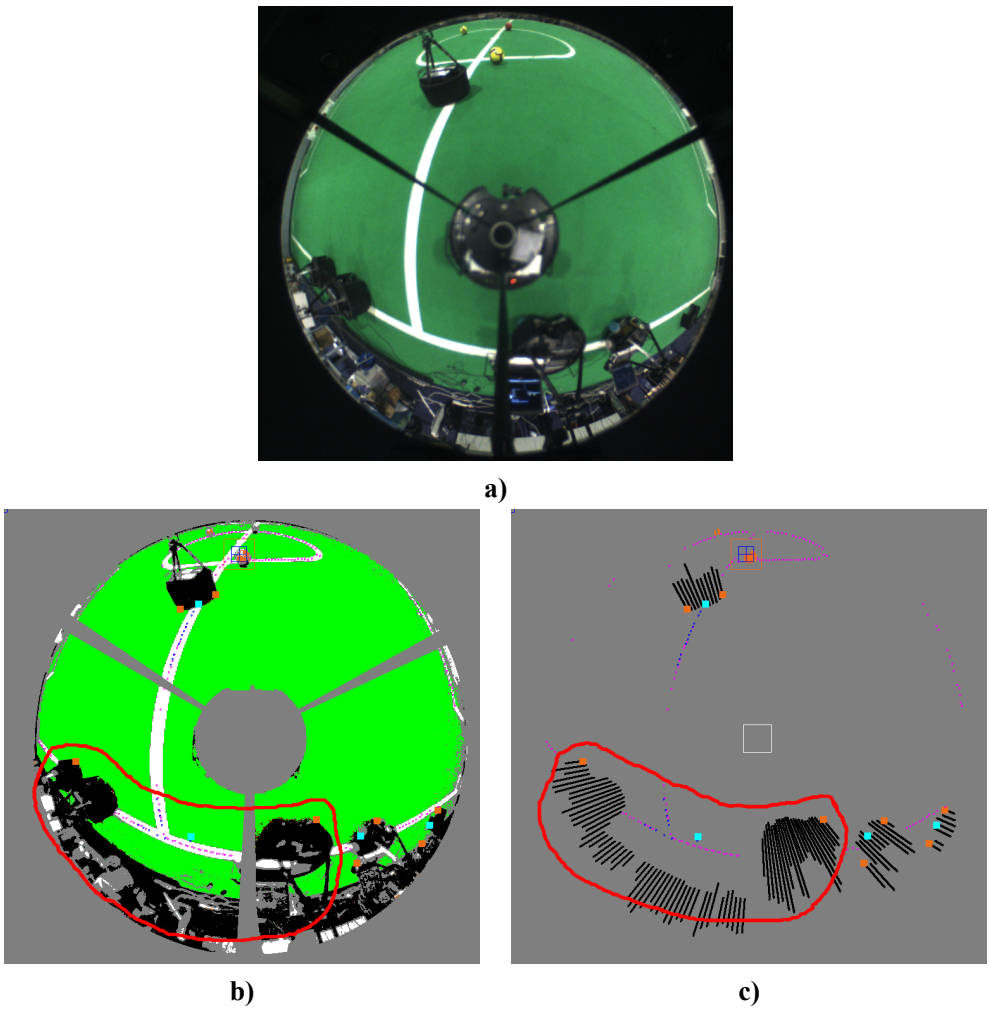
Figure 4: Relation between pixels and metric distances of the omnidirectional vision system. The center of the robot is considered the origin and the metric distances are considered on the ground plane.

Through the function represented in Figure 4, it is possible to create a normalized relation of blobs width and length with the distance. Sometimes an obstacle was separated in several blobs, mainly due to the noise in the image and problems in color classification, which led to fails in the detection of black regions in the scanlines. To avoid these situations, an offset was considered to decide when the angular space between blobs was considered enough to represent a real obstacle separation. The same principle was considered concerning the position of the black area in consecutive scanlines.

The separation offsets of a blob close to the robot were bigger than the ones at a high distance, to maintain coherent precision. The angular separation offset was considered for situations where robots were side-by-side, at the same distance, but there was no visual contact between each blob; the length separation offset was checked for situations where, on sequential scanlines, there were blobs with visual contact but the robots were actually at different distances. Other details can be found in (Silva et al., 2009).

The results of this approach were very dependent on a polynomial function for defining the pixel variation on the image. In practice it was verified that at higher distances (some tens of pixels from the mirror border) the offset definition in pixels was not accurate, as the distance relation tended to very high

values. Since the offset needs to depend on the inter-pixel distance, when maintaining an acceptably accurate threshold relation for shorter distances, the offset at longer distances would not be applicable. This caused the merging of blobs which were clearly part of different objects (Figure 5) or in some situations, caused clearly segmented obstacles to be ignored.



a)

b) **c)**

Figure 5: Capture of a camera frame with an incorrect visual obstacle detection as described. Top **a)**: Original frame; Bottom Left **b)**: Color segmented image; Bottom Right **c)**: Blob image obtained from segmentation. Obstacles are represented by the estimated visual limits (brown squares) and by its center (cyan square). The bottom left part of the image is merged within a single obstacle (surrounded by the red line).

Moreover, in the vision process, there is no information about the position of the robot and thus, from the vision process point of view, any black blob should be considered. However, we do not need to consider any obstacle that is outside the field, as it will not interfere with the game.

2.2 Obstacles detection as a sensor fusion problem

To avoid the situation previously described, a new way of determining the obstacles center and limits were implemented. Thanks to the known relation between pixel and distance, the relative coordinates of each point is known and can be made available for the integration process. Changes to the vision algorithm were made so that now, the position of each detected black point (the first black point on each scanline) is passed over to the integration process, instead of a visual estimation of the obstacle center and limits. The passed points are angularly ordered according to the scanlines angular definition.

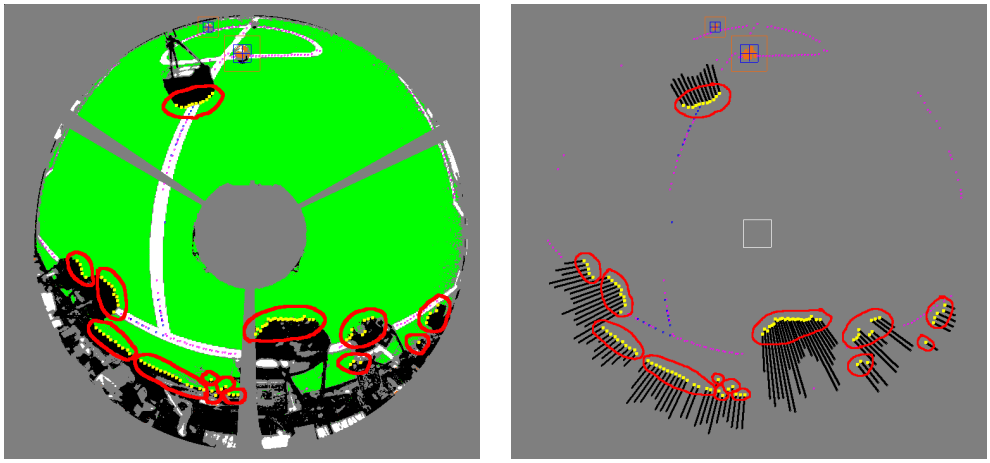
The integration process takes the points and builds the obstacles. This is made by an iterative process which starts by acquiring the first black point and define it as the limits of an obstacle. Iteratively, each black point is tested to be within a given neighborhood of the previous one. If it is within a given threshold metric distance, it is added to the currently obstacle being built and is assumed as the left limit. When a black point appears that does not belong to the neighborhood of the previous, the current obstacle is put on the obstacle list and a new obstacle is considered, starting with the current black point. After iterating all the points, the first and last obstacles limits are tested. If they are closer than the threshold, they are part of the same obstacle, which was divided by the start scanline of the search algorithm and thus they are merged (Listing 1).

```
1  temporaryObstacle.leftLimit = points[0];
2  temporaryObstacle.rightLimit = points[0];
3  for($p = 1$ to $totalNumberOfPoints-1$){
4     if(distance(points[p], points[p-1]) < threshold)
5        $temporaryObstacle.leftLimit = points[p];
6     }
7     else{
8        put temporaryObstacle on obstacle list;
9        temporaryObstacle.leftLimit = points[p];
10       temporaryObstacle.rightLimit = points[p];
11    }
12 }
13 last = numberOfCreatedObstacles - 1;
14 if(distance(obstacle[0].rightLimit, obstacle[last].leftLimit) <
       ↪ threshold){
15    merge limit obstacles;
16 }
```

Listing 1: Algorithm for merging visual points into obstacles.

This algorithm allows obstacles at longer distances to be evaluated in the same way as obstacles near the robot, without unfitting the threshold definition as was the case with pixel thresholds.

Additionally, since at the integration level the robot already knows its location, we can easily eliminate the black field borders or any other obstacle on the field vicinity by ignoring the obstacles that are outside the field by more than a given distance threshold. Having precision on the obstacle positions is very



a) **b)**

Figure 6: Capture of a camera frame with the new visual information provided by the vision process. Left **a)**: Color segmented image; Right **b)**: Blob image obtained from segmentation. All the detected valid black points are marked by yellow squares. All these points are provided to the integration which considers several obstacles (the red lines surround the obstacles created based on the described algorithm.

important for the next step of obstacle treatment, which is their selection and classification as team mates or opponents, described in Section 3.

In the same situation depicted in Figure 5, this new approach would visually provide all the points represented in Figure 6 (a) and (b) by the yellow squares and the result of the algorithm on the integration process would consider different obstacles, surrounded in the images by red lines. This separation of obstacles is more accurate than the previous pixel separation, as the integration process knows the context of the black points and the individual cartesian coordinates of each one. In our case, the cartesian distances are easier and more accurate to evaluate than pixel distances.

Note that the work described in the remaining of this chapter is independent of the vision algorithm, as long as the described information is available. Several approaches exist concerning obstacle identification. The work presented in (Lenser & Veloso, 2003) describes an approach for a similar application but with a non omni-directional vision system. In (Das et al., 2001; Koyasu et al., 2001), visual color and edge based detection algorithms are described, but both with times too high for use in the MSL environment. Other applications use several other sensors like laser range, stereo cameras or offline setup, but the financial and space costs of such solutions are much higher (Manduchi et al., 2005; Michels et al., 2005).

3 Obstacle Selection and Identification

With the objective of refining the information of the obstacles, and have more meaningfull and human readable information, the obstacles are selected and a cooperative matching is attempted, in order to try to identify them as team mates or opponents.

Due to the weaker precision at long distances, a first selection of the obstacles is made by selecting only the obstacles closer than a given distance as available for identification (currently 9 meters). Also, obstacles that are smaller than 10 centimeters wide or outside the field of play margin are ignored. This is done because the MSL robots are rather big, and in game situations small obstacles are not present inside the field. Also, it would be pointless to pay attention to obstacles that are outside the field of play, since the surrounding environment is completely ignorable for the game development.

To be able to distinguish obstacles, to identify which of them are team mates and which are opponent robots, a fusion between the own visual information of the obstacles and the shared team mates positions is made. By creating a circle around the team mate positions with the robot radius plus an error margin, varying with the distance, a matching of the estimated center of visible obstacle within the team mate area is made (Figure 7), and the obstacle is identified as the corresponding team mate in case of a positive matching (Figures 8 (c), 9(d)).

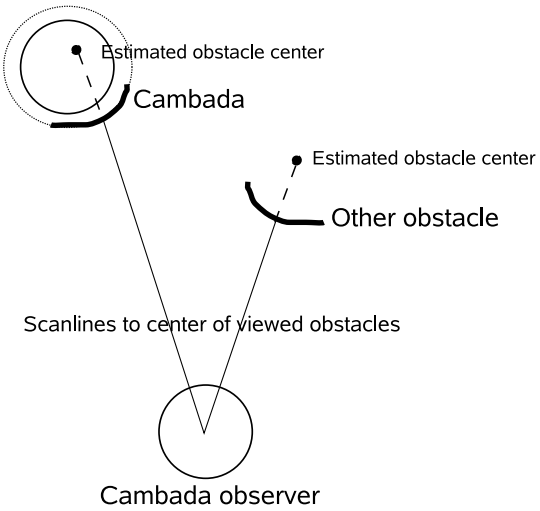


Figure 7: When a CAMBADA robot is on, the estimated centers of the detected obstacles are compared with the shared position of the team mates and tested if they are within the robot radius; the left obstacle is within the CAMBADA radius, the right one is not.

Since the obstacles detected can be large blobs, the above described identification algorithm cannot be applied directly to the visually detected obstacles. If the detected obstacle fulfills the minimum size requisites already described, it is selected as candidate for being a robot obstacle. Its size is evaluated and classified as robot if it does not exceed the maximum size allowed for MSL robots (MSL Technical Committee, 2010) (Figure 8a) and 8b)).

If the obstacle exceeds the maximum size of an MSL robot, a division of the obstacle is made, by analyzing its total size and verifying how many robots are in that obstacle. This is a common situation, robots clashing together and thus creating a compact black blob, originating a big obstacle (Figure 9a) and 9b)).

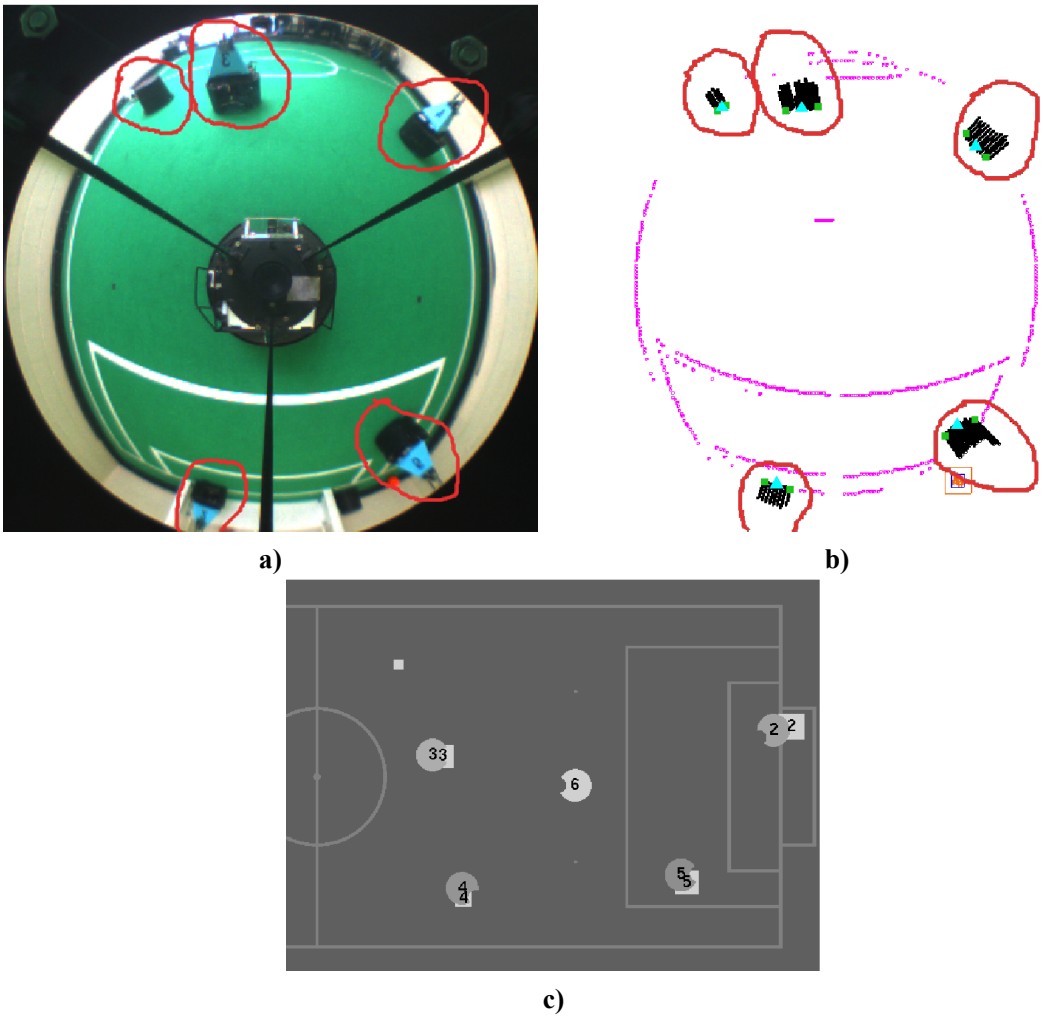


Figure 8: Illustration of single obstacles identification. Top Left **a)**: image acquired from the robot camera, denoting the single obstacles selected for identification (with lines surrounding them); Top Right **b)**: the same image after processing. The same obstacles are also denoted; Bottom **c)**: image of the control station, where each robot represents itself and the *observer* (robot 6, the lighter grey) draws all the 5 obstacles in evaluation conditions (represented by squares with the same grey scale as itself). All the obstacles correspondent to team mates were correctly identified (marked by its corresponding number over the obstacle square) and the opponent is also represented with no number. Note that the positions of the robots visible in the image (each has its number on the body) is inverted due to the mirrored vision system.

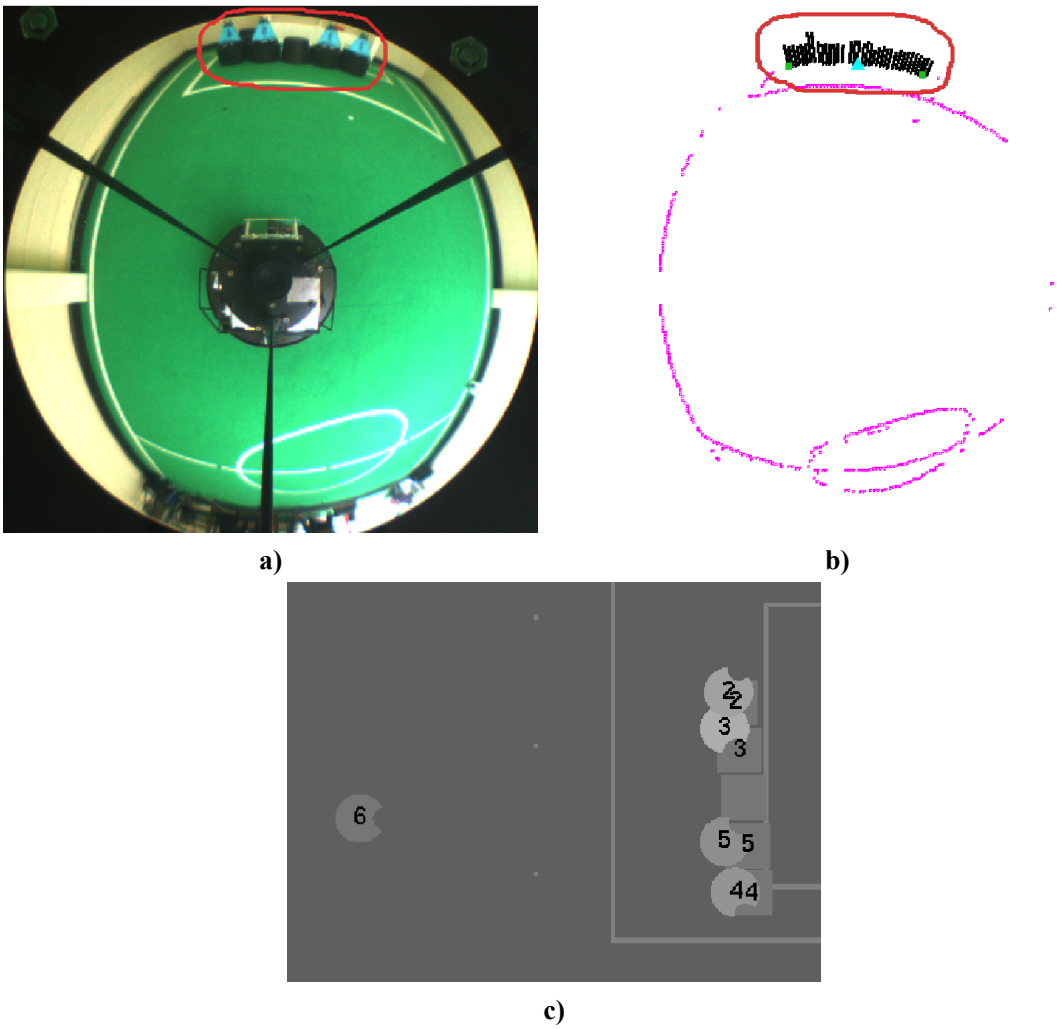


Figure 9: Illustration of multiple obstacles identification. Top Left **a)**: image acquired from the robot camera, denoting the obstacles aligned (with a line surrounding it); Top Right **b)**: the same image after processing. Visually, the aligned robots are only one large obstacle (marked with its center with a triangle and side limits with squares); Bottom **c)**: image of the control station, where each robot represents itself and the *observer* (robot 6, the darker grey) draws all the 5 obstacles (represented by squares with the same grey scale as itself). The visual obstacle was successfully separated into the several composing obstacles, and all of them were correctly identified as the correspondent team mate (marked by its corresponding number over the obstacle square) and the opponent is also represented with no number.

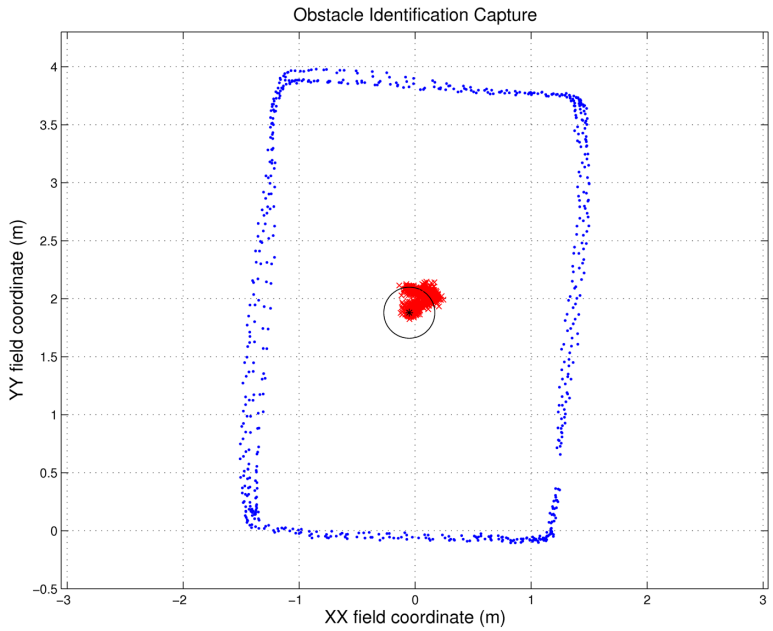


Figure 10: Representation of a capture of the obstacle identification algorithm results. The path taken by the *observer* is represented by blue dots in the rectangular path taken. Near the center, the *pivot* shared position is represented by the black star and its limits by the black circle. The blob of red is the overlapping positions of the identified obstacle center, represented by a red cross.

4 Experimental Results

The laboratory used for the tests receives natural light which can affect the vision processing algorithms. The presented results are not treated in any way to diminish the effects of natural light, as we are interested in understanding if the algorithms can cope with those conditions which can be found in real situations.

In the first test situation, a robot was positioned on the field at $(-0.05, 1.88)$ while broadcasting its position. This robot will be referred to as *pivot*. Another robot was moving on a rectangular path around the *pivot*, and a capture of its data was done. This robot will be referred to as *observer*. This scenario is intended to give some insight about the performance of the identification when the team mates are static or nearly static (as is the case of set plays during the games. In these situations it is important to analyze passing lines). Figure 10 is a graphic representation of the acquired data, with the *pivot* represented in black. The blue dots are the positions of the path taken by the *observer*, which covers the rectangular path for 3 times. In each cycle, the center of the obstacle perceived by the *observer* is represented by a red 'x'.

It is visible that, as expected, the obstacle position perceived by the *observer* is not exactly the *pivot* position. The capture in question is composed of 677 cycles. The identification of the obstacle as the correspondent team mate failed to succeed in only one cycle, which corresponds to a 99.85% success rate.

Considering that the *pivot* has 22 cm radius (although it is slightly bigger), the mean of the centers of the perceived obstacle is within the real area occupied by the *pivot*, at nearly 16 cm with a standard deviation

Perceived obstacle		
	X	**Y**
Mean	0.05	2.01
Std	0.08	0.07

$$|\vec{Real} - \vec{Perceived}| = 0.16$$
$$|Std| = 0.10$$

Table 1: Table with the mean and standard deviation of the capture perceived obstacle position

of 10 cm (Table 1).

Another test scenario was considered for evaluation of the algorithm performance for moving obstacles. Several captures were performed to evaluate the performance of the algorithm when identifying a moving team mate. This set of six captures consisted of a robot observing a team mate moving around and registering the data about the obstacles. The path taken by the moving team mate is represented in Figure 11. The number of failed identifications was greater when the moving robot was farther from the *observer*, as expected due to the noisy nature of the measurements.

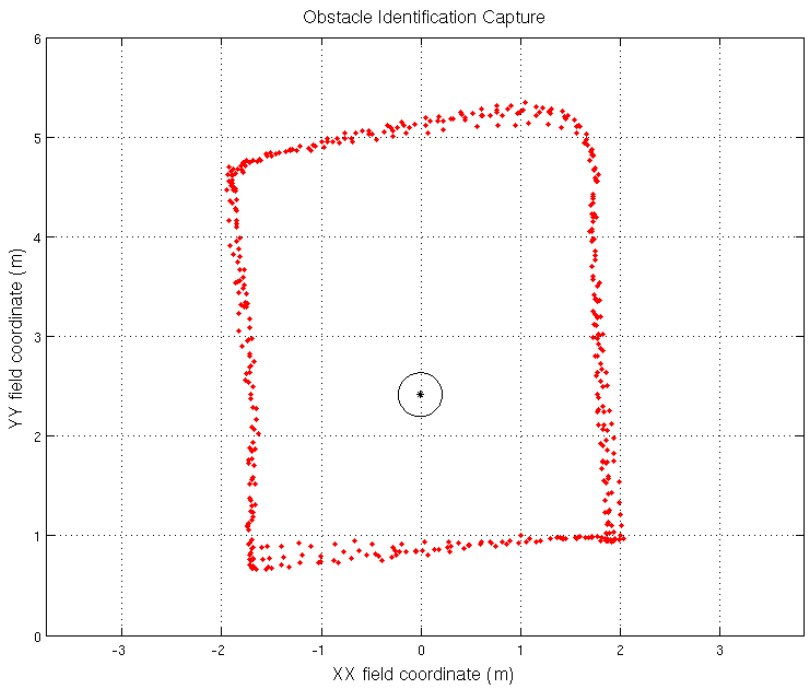


Figure 11: Representation of the path taken by the team mate to identify (the red dots represent each communicated position). The *observer* position is represented by the black star and its limits by the black circle.

The captures were performed throughout the day, with different lighting conditions but with the same robot calibration. Table 2 summarizes this set of captures, which revealed a total mean identification ratio of approximately 71%.

	Total cycles	Successes	%
Capture 1	1798	1319	73
Capture 2	1065	748	70
Capture 3	1528	1332	87
Capture 4	1162	769	66
Capture 5	1935	1278	66
Capture 6	2152	1411	66

Table 2: Table with the individual ratio of successful identification of the moving team mate for the several captures performed.

5 Obstacle Sharing

With the purpose of improving the global perception of the team robots, the sharing of locally known information is an important feature. Obstacle sharing allows the team robots to have a more global perception of the field occupancy, allowing them to estimate, for instance, passing and dribbling corridors more effectively.

However, one have to keep in mind that, mainly due to illumination conditions and eventual reflexive materials, some of the detected obstacles may not be exactly robots, but dark shadowy areas. If that is the case, the simple sharing of obstacles would propagate an eventually false obstacle among the team. Thus the algorithm for sharing the obstacles makes a fusion of the several team mates information.

The fusion of the information is done mate by mate. After building the worldstate by its own means, the agent checks all the team mates available, one by one. Their obstacles are matched with the own ones. If the agent does not know an obstacle shared by the team mate, it keeps it in a temporary list of unconfirmed obstacles. This is done to all the team mates obstacles. When another team mate shares a common obstacle, that same obstacle is confirmed and is transferred to the local list of obstacles. In the current cycle, the temporary obstacles that were not confirmed are not considered. An outline of the algorithm is presented in Listing 2.

The matching of the team mate obstacles with the own obstacles is done in a way similar to the matching of the obstacle identification with the team mate position described in section 3. The cambada *pivot* in Figure 7 is replaced by the current team mate obstacle for the matching test.

Figure 12 shows a situation where robot 2, in the goal area was too far to see the obstacle on the middle of the field. Thus, it considered the obstacle in question, only because it is identified by both robots 5 and 6, as visible in the figure.

```
for(c=1 to totalNumberOfTeamMates){
   for(o=1 to totalObstaclesOfTeamMate){
      for(m=1 to totalOwnObstacles){
         if(m matches o){
            Obstacle already known, do nothing
         }
         else{
           if(m was previously communicated by another team mate}){
              temporary obstacle confirmed and added;
           }
           else{
              obstacle considered temporarily;
              waits for confirmation by another team mate;
            }
         }
      }
   }
}
```

Listing 2: Algorithm for obstacle sharing.

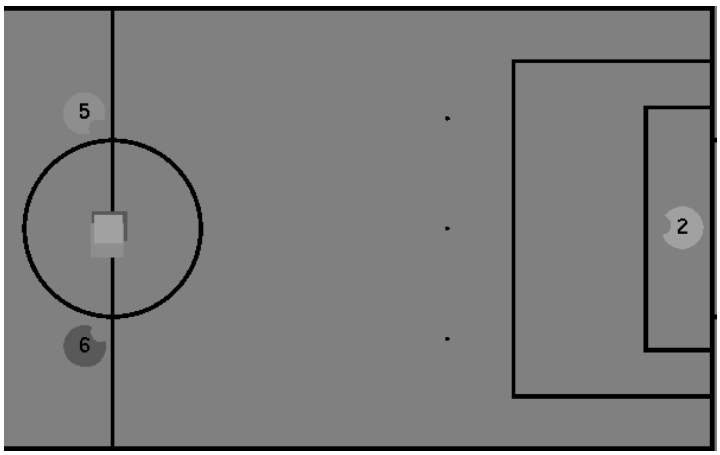


Figure 12: Image of the control station showing an obstacle of robot 2 that was not seen by itself (on the centre of the field). In this case it assumes the obstacle by confirmation of both robots 5 and 6.

6 Conclusion

The integration of obstacles in the robot representation of the world is an important issue for playing the game. The described treatment of obstacles, used in the last competitions, has been monitored by observation of the control station. We verified an increase of obstacle detection, stability and coherence among team mates. The identification of obstacles provided a good tool for pass evaluation, since it allows the strategic layer to easily verify if the corridor needed for a pass is free of opponents, while ignoring team mate obstacles near the corridor, as they would not intercept the pass.

Also, one of the main ways we wish to take advantage of the improved obstacle model is the implementation of a path planning strategy by the strategic layer software.

Improvements are being implemented to allow to create a global and unique identification also for the opponent robots. This will be achieved by keeping a history of each and every obstacle so it can be tracked, while making periodic updates of shared information, so all the robots of the team can synchronize and unify the identification of their opponents periodicaly.

The new representation of obstacles in the CAMBADA world model has thus been effective and important to the team performance during the last competitions, but it has also provided the base for more developments in several areas to allow improvements.

Acknowledgments

This work has been supported by the Instituto de Engenharia Electrónica e Telemótica de Aveiro (IEETA), the Universidade de Aveiro (UA) and Fundação para a Ciência e Tecnologia (FCT).

References

Cunha, B., Azevedo, J., Lau, N., & Almeida, L. (2008). Obtaining the inverse distance map from a non-svp hyperbolic catadioptric robotic vision system. In U. Visser, F. Ribeiro, T. Ohashi, & F. Dellaert (Eds.), RoboCup 2007: Robot Soccer World Cup XI, volume 5001 of Lecture Notes in Artificial Intelligence (pp. 417–424).: Springer.

Das, A. K., Fierro, R., Kumar, V., Southall, B., Spletzer, J., & Taylor, C. J. (2001). Real-time vision-based control of a nonholonomic mobile robot. In IEEE International Conference on Robotics and Automation (pp. 1714–1719). Seoul, Korea.

Kitano, H., Asada, M., Kuniyoshi, Y., Noda, I., & Osawa, E. (1997). RoboCup: The robot world cup initiative. In Proceedings of the first international conference on Autonomous agents (pp. 340–347).: ACM New York, NY, USA.

Koyasu, H., Miura, J., & Shirai, Y. (2001). Realtime omnidirectional stereo for obstacle detection and tracking in dynamic environments. IEEE International Conference on Intelligent Robots and Systems, (pp. 31–36).

Lenser, S. & Veloso, M. (2003). Visual sonar: Fast obstacle avoidance using monocular vision. In IEEE International Conference on Intelligent Robots and Systems (pp. 886–891). Las Vegas, Nevada.

Manduchi, R., Castano, A., Talukder, A., & Matthies, L. (2005). Obstacle detection and terrain classification for autonomous off-road navigation. Autonomous Robots, (pp. 81–102).

Michels, J., Saxena, A., & Ng, A. Y. (2005). High speed obstacle avoidance using monocular vision and reinforcement learning. Proceedings of the 22nd International Conference on Machine Learning.

MSL Technical Committee (2010). Middle Size Robot League Rules and Regulations for 2011.

Neves, A., Corrente, G., & Pinho, A. (2007). An omnidirectional vision system for soccer robots. In Progress in Artificial Intelligence, volume 4874 of Lecture Notes in Artificial Intelligence (pp. 499–507). Springer.

Neves, A., Martins, D., & Pinho, A. (2008). A hybrid vision system for soccer robots using radial search lines. In Proc. of the 8th Conference on Autonomous Robot Systems and Competitions, Portuguese Robotics Open - Robótica 2008 (pp. 51–55). Aveiro, Portugal.

Silva, J., Lau, N., Neves, A. J. R., ao Rodrigues, J., & Azevedo, J. L. (2009). Obstacle detection, identification and sharing on a robotic soccer team. In Portuguese Conference on Artificial Intelligence (EPIA), volume 5816 (pp. 350–360). Aveiro, Portugal.