AN EFFICIENT REAL-TIME HYBRID VISION SYSTEM FOR SOCCER ROBOTS

António J. R. Neves, Daniel A. Martins, Armando J. Pinho and José Luís Azevedo

Transverse Activity on Intelligent Robotics, IEETA / DETI
University of Aveiro, 3810–193 Aveiro, Portugal
an@ua.pt — dam@ua.pt — ap@ua.pt — jla@ua.pt

ABSTRACT

In many robotic applications, autonomous robots must be capable of locating the objects that they have to manipulate. In the case of autonomous soccer robots, they must, at least, be able to locate the ball, the opponent robots and the team robots, and also to collect field information essential for self-localization. The recognition of colored objects is very important for robot vision in RoboCup Middle Size League competition. This paper describes an efficient hybrid vision system developed for the robotic soccer team of the University of Aveiro, CAMBADA (Cooperative Autonomous Mobile roBots with Advanced Distributed Architecture). The hybrid vision system integrates an omnidirectional and a perspective camera. The omnidirectional sub-system is used by our localization algorithm for finding the ball, detecting the presence of obstacles and white lines. The perspective vision is used to find the ball and obstacles in front of the robot at larger distances, which are difficult to detect using the omnidirectional vision system. In this paper, we present a set of algorithms for efficiently extracting the color information of the acquired images and, in a second phase, for extracting the information of all objects of interest. We developed an efficient color extraction algorithm based on lookup tables and we use a radial model for object detection, both in the omnidirectional and perspective sub-system. The CAMBADA middle-size robotic soccer team won the 2008 RoboCup World Championship and the 2007 and 2008 Portuguese Robotics Festival. These results show the effectiveness of our algorithms. Moreover, our experiments show that the system is fast and accurate having a constant processing time independently of the environment around the robot, which is a desirable property of Real-Time systems.

1. INTRODUCTION

Vision is an extremely important sense for both humans and robots, providing detailed information about the environment. A robust vision system should be able to detect objects reliably and provide an accurate representation of the world to higher level processes. The vision system must also be highly efficient, allowing a resource-limited agent to respond quickly to a changing environment. Each frame acquired by a digital camera must be processed in a small, usually fixed, amount of time. Algorithmic complexity is therefore constrained, introducing a trade-off between processing time and the quality of the information acquired.

The Middle Size League (MSL) competition of RoboCup is a standard real-world test for autonomous multi-robot control. Being yet a color-coded environment, despite the recent changes introduced, such as the goals without color, recognizing colored objects such as the orange ball, the black obstacles, the green field and the white lines are a basic ability for robots.

One problem domain in RoboCup is the field of Computer Vision, responsible for providing basic information that is needed for calculating the behavior of the robots. Catadioptric vision systems (often named omnidirectional vision systems) have captured much interest in the last years, because they allow a robot to see in all directions at the same time with having to move itself or its camera [1, 2, 3, 4, 5]. However, due to the last changes in the MSL rules, the playing field became larger, bringing some problems to the omnidirectional vision systems, particularly regarding the detection of objects at large distances.

The main goal of this paper is to present an efficient real-time hybrid vision system for the world champion robotic soccer team of the University of Aveiro, CAMBADA (see Fig. 1). We propose an hybrid vision system to process the video acquired by an omnidirectional camera and a perspective camera. The system finds the white lines of the playing field (used for self-localization), the ball and obstacles. Our vision system architecture uses a distributed paradigm where the main tasks, namely, image acquisition, color extraction, object detection and image visualization, are separated into several processes, as presented in Fig. 2.

Fig. 1. Robots used by the CAMBADA middle-size robotic soccer.

The image processing software uses radial search lines to analyze the color information. A radial search line is a line that starts at the center of the robot, with some angle, and ends at the limit of the image. The center of the robot in the omnidirectional subsystem is approximately the center of the image. However, the center of the robot in the perspective subsystem is at the bottom of the image. For each search line, if it is found a predefined number of pixels classified as a valid color, the system saves the position of the first pixel associated to the respective color. For finding the white lines, color transitions from green to white are searched for.

This work was supported in part by the Fundação para a Ciência e a Tecnologia (FCT).

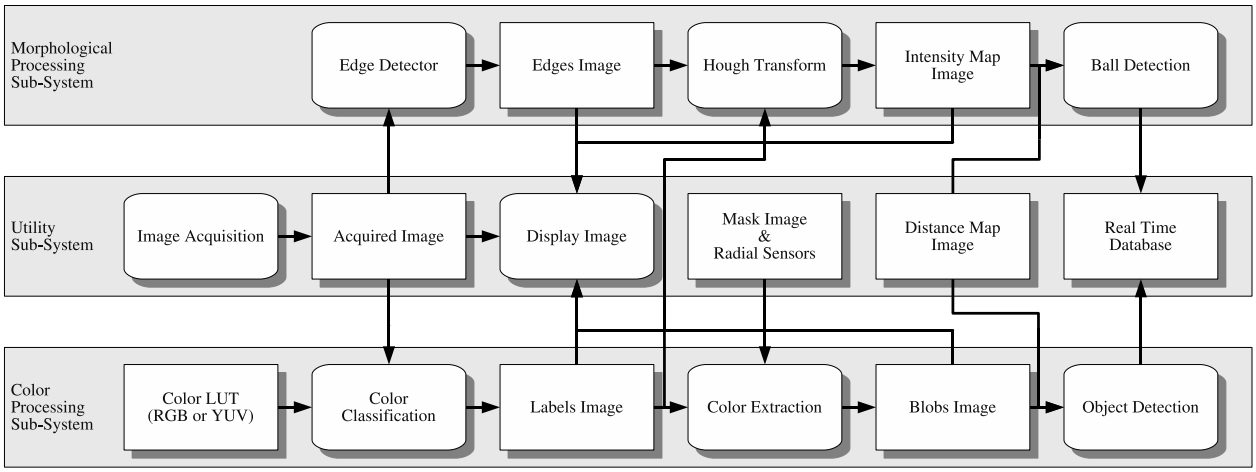

Fig. 2. The architecture of the vision system, applied both to the omnidirectional and perspective subsystem.

A lookup table (LUT) is used for color classification. Our system is prepared to acquire images in RGB 24-bit, YUV 4:2:2 or YUV 4:1:1 format, being necessary only to choose the appropriated LUT. However, we use the HSV color space for color calibration due to its special characteristics [6].

This paper is organized as follows. In Section 2 we describe the general architecture of the CAMBADA robots. Section 3 presents our vision system architecture, explaining the several modules developed and how they are connected. Section 4 presentes the algorithms used to collect the color information of the images using radial search lines. In Section 5 we show how the object features are extracted and some experimental results are also presented. Finally, in Section 6, we draw some conclusions.

2. ARCHITECTURE OF THE ROBOTS

The CAMBADA robots (Fig. 1) were designed and completely built in-house. The baseline for robot construction is a cylindrical envelope, with 485 mm in diameter. The mechanical structure of the players is layered and modular. Each layer can easily be replaced by an equivalent one. The components in the lower layer, namely motors, wheels, batteries and an electromagnetic kicker, are attached to an aluminum plate placed 8 cm above the floor. The second layer contains the control electronics. The third layer contains a laptop computer, at 22.5 cm from the floor, an omni-directional vision system, a frontal camera and an electronic compass, all close to the maximum height of 80 cm. The players are capable of holonomic motion, based on three omni-directional roller wheels.

The robots computing system architecture follows the fine-grained distributed model [7] where most of the elementary functions, e.g. closed loop control of complex actuators, are encapsulated in small microcontroller based nodes, connected through a network. A laptop node is used to execute higher-level control functions and to facilitate the interconnection of off-the-shelf devices, e.g. cameras, through standard interfaces, e.g. USB or Firewire. For this purpose, Controller Area Network (CAN), a real-time fieldbus typical in distributed embedded systems, has been chosen. This network is complemented with a higher-level transmission control protocol to enhance its real-time performance, composability and fault-tolerance, namely the FTT-CAN protocol (Flexible Time-Triggered communication over CAN) [8].

The communication among robots and to the base station uses the standard wireless LAN protocol IEEE 802.11x profiting from large availability of complying equipment.

The software system in each player is distributed among the various computational units. High-level functions run on the computer, a laptop PC running Linux operating system. Low-level functions run on dedicated microcontrollers. A cooperative sensing approach based on a Real-Time Database (RTDB) has been adopted [9]. The RTDB is a data structure where players share their world models. It is updated and replicated in all players in real-time.

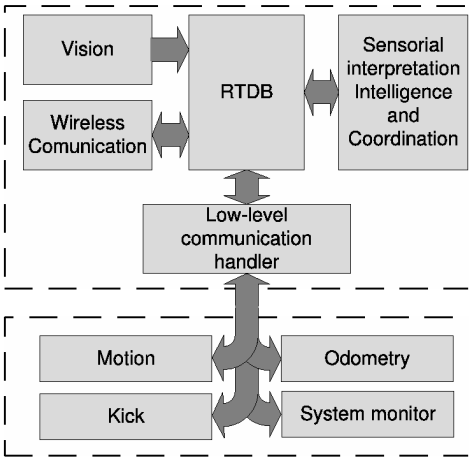

Fig. 3. Layered software architecture of CAMBADA players [7].

A software module called `Integrator` is used to update the world state information [10]. This is done by filtering the raw information coming from sensors (i.e. vision, odometry, etc.) and determining the best estimate of the position and velocity of each object.

The software of CAMBADA players is composed of several different processes that have responsibility for different tasks: image acquisition, image analysis, integration/decision and communication with the low-level modules. The order and schedule of activation of

these processes is performed by a process manager library called `Pman`. `Pman` stores in a database the characteristics of each process to activate and allows the activation of recurrent tasks, settling phase control (through the definition of temporal offsets), precedence restrictions, priorities, etc. The `Pman` services allow changes in the temporal characteristics of the process schedule during runtime [11, 12].

It is very important that all robots share the same play mode obtained by processing the referee orders given through the referee box. In CAMBADA, an application inside the team's base station checks the messages received from the referee box, and converts the event triggered protocol of communication referee box - base station to a state oriented playmode information that is broadcasted to robots using the RTDB. This ensures that the delay between the reception of a referee event from the referee box and its awareness by all robots is minimized, enabling a synchronized collective behavior. The general architecture of the CAMBADA robots (see Fig 1) has been described in [7]. Basically, the robot architecture is centered on a main processing unit that is responsible for the higher-level behavior coordination, i.e. the coordination layer [13]. This main processing unit (a laptop) processes visual information gathered from the vision system, executes high-level control functions and handles external communication with the other robots. This unit also receives sensing information and sends actuating commands to control the robot attitude by means of a distributed low-level sensing/actuating system. The PC runs the Linux operating system. The communication among team robots uses an adaptive TDMA transmission control protocol [14] on top of IEEE 802.11b, that reduces the probability of transmission collisions between team mates thus reducing the communication latency. Using this transmission protocol a Shared Real Time Database (RTDB) is implemented.

3. HYBRID VISION SYSTEM

This paper presents a hybrid vision system integrating an omnidirectional and a perspective camera. The omnidirectional part of the vision system [15] is based on a catadioptric configuration implemented with a firewire camera (with a 1/3" CCD sensor and a 4.0mm focal distance lens) and a hyperbolic mirror. The perspective camera uses a low cost firewire camera (BCL 1.2 Unibrain camera with a 1/4" CCD sensor and a 3.6mm focal distance lens). The omnidirectional camera works at 30 frames per second (fps) in YUV4:2:2 mode with a resolution of 640×480 pixels. The front camera works at 30 fps in YUV4:1:1 mode with a resolution of 640×480 pixels.

The omnidirectional system is used to find the ball, to detect the presence of obstacles and white lines. The perspective vision is used to find the ball and obstacles in front of the robot at larger distances, which are difficult to detect using the omnidirectional vision system.

The software architecture is based on a distributed paradigm grouping main tasks in different modules. The software can be split in three main modules, namely the *Utility Sub-System*, the *Color Processing Sub-System* and the *Morphological Processing Sub-System*, as can be seen in Fig. 2. Each one of these sub-systems labels a domain area where their processes fit, as the case of *Acquire Image* and *Display Image* in the *Utility Sub-System*. As can be seen in *Color Processing Sub-System*, proper color classification and extraction processes were developed, along with an object detection process to extract information, through color analysis, from the acquired image [16, 15]. The *Morphological Processing Sub-System*, presented in [17], are used to detect arbitrary FIFA balls, independent of their color.

Despite the obvious differences between the omnidirectional and the perspective sub-systems, the software architecture used in both is the same, changing only the *Image Mask & Radial Sensors* and the *Distance Mapping Image* [16].

4. COLOR EXTRACTION

Image analysis in the RoboCup domain is simplified, since objects are color coded. Black robots play with an orange ball on a green field that has white lines. Thus, the color of a pixel is a strong hint for object segmentation. We exploit this fact by defining color classes, using a look-up table (LUT) for fast color classification. The table consists of $16\,777\,216$ entries (2^{24}, 8 bits for red, 8 bits for green and 8 bits for blue), each 8 bits wide, occupying 16 MBytes in total. If another color space is used, the table size is the same, changing only the "meaning" of each component. Each bit expresses whether the color is within the corresponding class or not. This means that a certain color can be assigned to several classes at the same time. To classify a pixel, we first read the pixel's color and then use the color as an index into the table. The value (8 bits) read from the table will be called "color mask" of the pixel.

The color calibration is performed in HSV (Hue, Saturation and Value) color space due to its special characteristics. In the current setup, the image is acquired in RGB or YUV format and then is converted to an image of labels using the appropriate LUT.

There are certain regions in the received images that have to be excluded from analysis. Regarding the omnidirectional camera, one of them is the part in the image that reflects the robot itself. Other regions are the sticks that hold the mirror and the areas outside the mirror. For that, we have an image with this configuration that is used by our software. An example is presented in Fig. 4. The white pixels are the area that will be processed, black pixels will not. Regarding the perspective camera, there are two regions that will not be processed. One at the top of the image, where the objects (ball and obstacles) are outside the field, and another at the bottom of the image, where the ball and obstacles are easily detected by the omnidirectional camera. With this approach we can reduce the time spent in the conversion and searching phases and we eliminate the problem of finding erroneous objects in that areas.

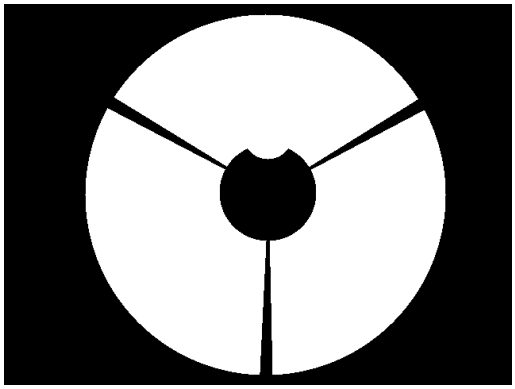

Fig. 4. An example of a robot mask used to select the pixels to be processed by the omnidirectional vision sub-system. White points represent the area that will be processed.

To extract the color information of the image we use radial search lines instead of processing the whole image. A radial search line is a line that starts at the center of the robot, with some angle, and ends at the limit of the image. The center of the robot in the

omnidirectional subsystem is approximately the center of the image (Fig. 5). In the perspective subsystem, the center of the robot is located at the bottom of the image (Fig. 6). The search lines are constructed based on the Bresenham line algorithm [18]. These search lines are constructed once, when the application starts, and saved in a structure in order to improve the access to these pixels in the color extraction module. For each search line, we iterate through its pixels to search for two things: transitions between two colors and areas with specific colors.

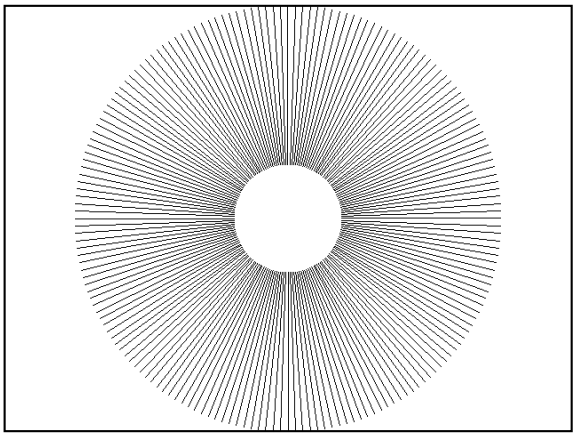

Fig. 5. The position of the radial search lines used in the omnidirectional vision sub-system.

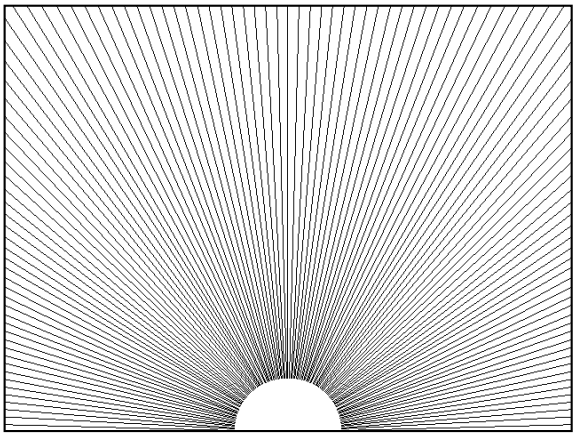

Fig. 6. The position of the radial search lines used in the perspective vision sub-system.

The use of radial search lines accelerates the process of object detection due to the fact that we only process about 30% of the valid pixels. This approach has a processing time almost constant, independently of the information around the robot, being a desirable property in Real-Time Systems. This happens because the system processes almost the same number of pixels in each frame. Regarding the omnidirectional vision sub-system, there is another advantage due to the fact that the use of omnidirectional vision difficult the detection of the objects using, for example, their bounding box. In this case, it is more desirable to use the distance and angle, which are inherent to the radial search lines.

We developed an algorithm to detect areas of a specific color which eliminates the possible noise that could appear in the image. For each radial scanline, it is performed a median filtering procedure as described next. Each time that a pixel is found with a color of interest, we analyze the pixels that follows (a predefined number) and if we don't find more pixels of that color we discard the pixel found and continue. When we find a predefined number of pixels with that color, we consider that the search line has this color.

In order to improve the previous algorithm, we created an algorithm to recover lost orange pixels due to the ball shadow cast over itself. As soon as we find a valid orange pixel in the radial sensor, the shadow recovery algorithm tries to search for darker orange pixels previously discarded in the color segmentation analysis. This search is conducted, in each radial sensor, starting in the first orange pixel found in direction to the center of the robot, limited to a maximum number of pixels. For each pixel analyzed, a wide color space comparison is performed in order to accept darker orange pixels. Once a different color is found or the maximum number of pixels is reached, the search in the current sensor is completed and the search proceeds to the next sensor. In Fig. 10 we can see the pixels recovered by this algorithm (orange pixels in the original image).

To accelerate the process of calculating the position of the objects, we put the color information found in each search line into a list of colors. We are interested in the first pixel (in the corresponding search line) where the color was found and the number of pixels with that color that have been found in the search line. Then, using the previous information, we separate the information of each color into blobs (Fig. 10 and 11 shows some examples). After this, it is calculated the blob descriptor that will be used for the object detection module, which consists in the following information:

- Average distance to the robot;
- Mass center;
- Angular width;
- Number of pixels;
- Number of green pixels between blob and the robot;
- Number of pixels after blob.

5. OBJECT DETECTION

The objects of interest that are present in the RoboCup environment are: a ball, obstacles (other robots) and the green field with white lines. Currently, our system detects efficiently all these objects with a set of simple algorithms that, using the color information collected by the radial search lines, calculate the object position and / or their limits in an angular representation (distance and angle).

The algorithm that searches for the transitions between green pixels and white pixels is described as follows. If a non-green pixel is found, we will look for a small window in the "future", and count the number of non-green pixels and of white pixels. Next, we look for a small window in the "past" and a small window in the "future" and count the number of green pixels. If these values are greater than a predefined threshold, we consider this point as a transition. This algorithm is illustrated in Fig. 7.

The transition points detected are used for the robot localization. All the points detected are sent to the Real-time Database, afterward used by the localization process.

To detect the ball, we use the following algorithm:

1. Separate the orange information into blobs.
2. For each blob, calculate the information described in the previous section.

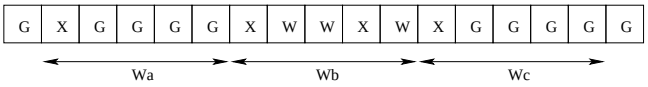

Fig. 7. An example of a transition. "G" means green pixel, "W" means white pixel and "X" means pixel with a color different from green or white.

3. Perform a first validation of the orange blobs using the information about the green pixels after and before the blob.
4. Validate the remain orange blobs according to the experimental data illustrated in Fig. 8 and 9.
5. Using the history of the last ball positions, choose the best candidate blob. The position of the ball is the mass center of the blob.

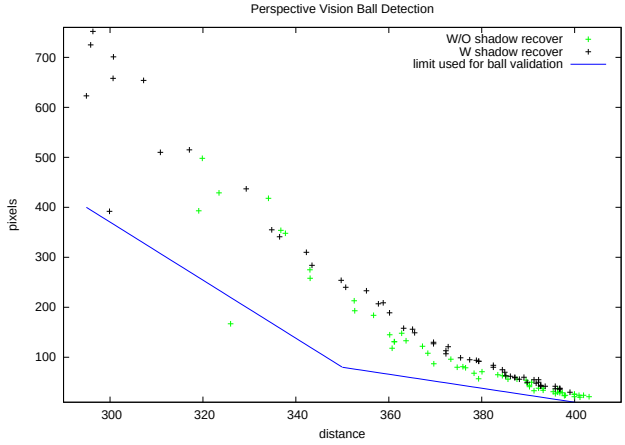

Fig. 8. Experimental results comparing the number of pixels for the ball according to the distance to the robot for the perspective camera.

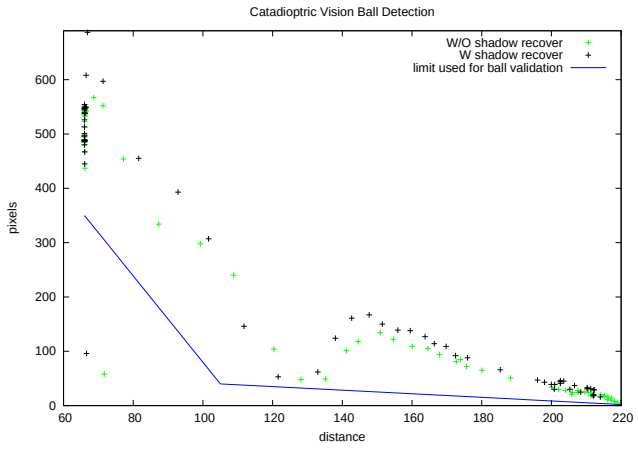

Fig. 9. Experimental results comparing the number of pixels for the ball according to the distance to the robot for the omnidirectional camera.

In order to avoid false positive errors in ball detection, a ball validation system was implemented. Experimental data regarding the number of pixels of a ball as a function of its distance to the robot is displayed in Figs. 8 and 9. Based on these data, we established a minimum limit of pixels, varying linearly with the distance, that an orange blob should have in order to be considered a ball candidate. The behavior of the acquired data, for the omnidirectional camera, around distance 120, is due to some occlusion of the ball by the mirror holding structure. This method, besides simple, is robust, fast and easy to implement.

To calculate the position of the obstacles around the robot, we use the following algorithm:

1. Separate the black information into blobs.
2. If the angular width of one blob is greater than 10 degrees, split the blob into smaller blobs, in order to obtain an accurate information about obstacles.
3. Calculate the information for each blob.
4. The position of the obstacle is given by the distance of the blob relatively to the robot. The limits of the obstacle are obtained using the angular width of the blob.

In Figs. 10 and 11 we present some examples of acquired images, their correspondent segmented images and the detected color blobs. As we can see, the objects are correctly detected (see the marks in the images on the right of the figures).

The proposed hybrid vision system has a constant processing time, independently of the environment around the robot, rounding 10 ms for the omnidirectional subsystem and 6 ms for the perspective subsystem, being the most part of the time spent in the color classification and in the color extraction modules. Each of the subsystems needs approximately 30 MBytes of memory. These results have been obtained using a laptop with an Intel Core 2 duo at 2.0 GHz and 1 GB of memory.

6. CONCLUSIONS

This paper presents the hybrid vision system developed for the CAMBADA middle-size robotic soccer team. The hybrid vision system integrates an omnidirectional and a perspective camera. We presented several algorithms for image acquisition and processing. The CAMBADA team won the 2007 and 2008 Portuguese Robotics Open and ranked 5th in the 2007 RoboCup World Championship. More recently, the CAMBADA team won the 2008 RoboCup World Championship, demonstrating the effectiveness of the proposed architecture in a competition environment.

Being a color coded environment, recognizing colored objects in the context of the RoboCup MSL competition, such as the orange ball, the black obstacles, the green field and the white lines, are a basic ability for robots. Therefore, our system defines different color classes corresponding to the objects. The 24 bit pixel color is used as an index to a 16 MBytes lookup table which contains the classification of each possible color in a 8 bit entry. Each bit specifies whether that color lays within the corresponding color class.

The processing system is divided into two phases: color extraction, using radial search lines, and object detection, using specific algorithms. The objects involved are: a ball, obstacles and white lines. The processing time and the accuracy obtained in the object detection confirms the effectiveness of our system.

The proposed system is still under development, in particular with the aim of recognizing objects using morphological information. In the RoboCup MSL the color codes tend to disappear. The color of the ball is the next color scheduled to disappear. Therefore, in [17] we proposed a solution to detect balls independently of their

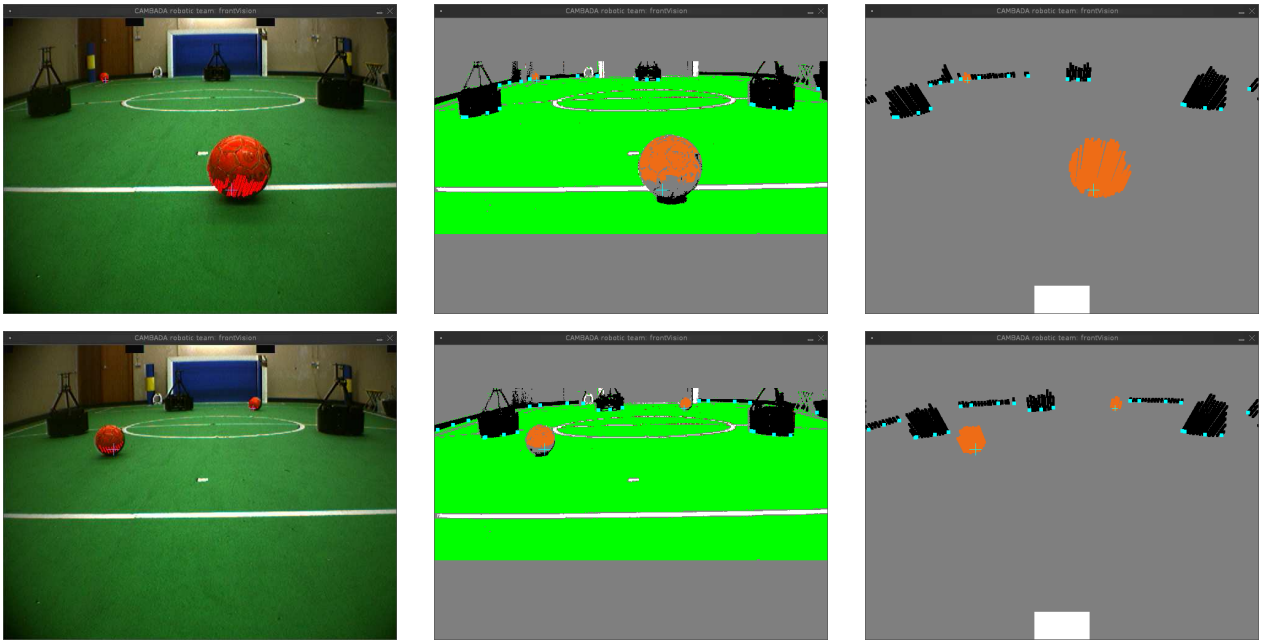

Fig. 10. On the left, examples of original images acquired by the perspective vision sub-system. In the center, the corresponding image of labels. On the right, the color blobs detected in the images. Marks over the ball point to the mass center. The cyan marks are the position of the obstacles.

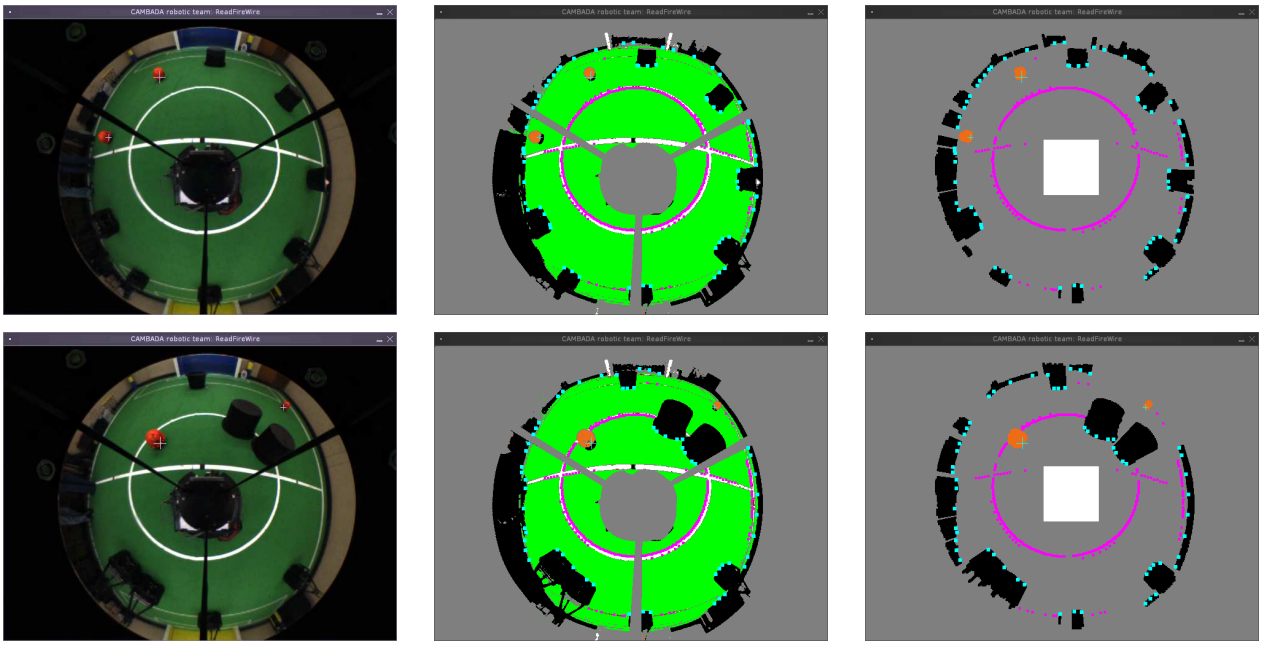

Fig. 11. On the left, examples of original images acquired by the omnidirectional vision sub-system. In the center, the corresponding image of labels. On the right, the color blobs detected in the images. Marks over the ball point to the mass center. The several marks near the white lines (magenta) are the position of the white lines. The cyan marks are the position of the obstacles.

color. This solution is based on a morphological analysis of the image, being strictly directed to detect round objects in the field, in this case the ball. Although the method proposed in [17] is under development, the preliminary experimental results are very encouraging. Using this image processing system, the CAMBADA team ranked 2nd in the mandatory challenge at RoboCup 2008, where the robots should play with an arbitrary standard FIFA ball.

7. REFERENCES

[1] Z. Zivkovic and O. Booij, "How did we built our hyperbolic mirror omni-directional camera - practical issues and basic geometry," Tech. Rep., Intelligent Systems Laboratory, University of Amsterdam, 2006.

[2] J. Wolf, "Omnidirectional vision system for mobile robot localization in the robocup environment," M.S. thesis, Graz University of Technology, 2003.

[3] E. Menegatti, F. Nori, E. Pagello, C. Pellizzari, and D. Spagnoli, "Designing an omnidirectional vision system for a goalkeeper robot," in *Proc. of RoboCup 2001*. 2001, vol. 2377 of *Lecture Notes in Computer Science*, pp. 78–87, Springer.

[4] E. Menegatti, A. Pretto, and E. Pagello, "Testing omnidirectional vision-based monte carlo localization under occlusion," in *Proc. of the IEEE Intelligent Robots and Systems, IROS 2004*, 2004, pp. 2487–2493.

[5] P. Lima, A. Bonarini, C. Machado, F. Marchese, C. Marques, F. Ribeiro, and D. Sorrenti, "Omni-directional catadioptric vision for soccer robots," *Robotics and Autonomous Systems*, vol. 36, no. 2-3, pp. 87–102, 2001.

[6] P. M. R. Caleiro, A. J. R. Neves, and A. J. Pinho, "Color-spaces and color segmentation for real-time object recognition in robotic applications," *Revista do DETUA*, vol. 4, no. 8, pp. 940–945, June 2007.

[7] J. L. Azevedo, B. Cunha, and L. Almeida, "Hierarchical distributed architectures for autonomous mobile robots: a case study," in *Proc. of the 12th IEEE Conference on Emerging Technologies and Factory Automation, ETFA2007*, Greece, 2007, pp. 973–980.

[8] L. Almeida, P. Pedreiras, and J. A. Fonseca, "The FTT-CAN protocol: Why and how," *IEEE Transactions on Industrial Electronics*, vol. 49, no. 6, pp. 1189–1201, 2002.

[9] L. Almeida, F. Santos, T. Facchinetti, P. Pedreira, V. Silva, and L. S. Lopes, "Coordinating distributed autonomous agents with a real-time database: The CAMBADA project," in *Proc. of the 19th International Symposium on Computer and Information Sciences, ISCIS 2004*. 2004, vol. 3280 of *Lecture Notes in Computer Science*, pp. 878–886, Springer.

[10] J. Silva, N. Lau, J. Rodrigues, and J. L. Azevedo, "Ball sensor fusion and ball interception behaviours for a robotic soccer team," in *Proc. of the 11th edition of the Ibero-American Conference on Artificial Intelligence, IBERAMIA 2008*, Lisbon, Portugal, october 2008.

[11] P. Pedreiras, F. Teixeira, N. Ferreira, L. Almeida, A. Pinho, and F. Santos, "Enhancing the reactivity of the vision subsystem in autonomous mobile robots using real-time techniques," in *Proc. of RoboCup 2006*. 2006, vol. 4020 of *Lecture Notes in Computer Science*, pp. 371–383, Springer.

[12] P. Pedreiras and L. Almeida, *Task Management for Soft Real-Time Applications Based on General Purpose Operating Systems, Robotic Soccer*, Itech Education and Publishing, Vienna, Austria, 2007.

[13] N. Lau, L. S. Lopes, and G. Corrente, "CAMBADA: information sharing and team coordination," in *Proc. of the 8th Conference on Autonomous Robot Systems and Competitions, Portuguese Robotics Open - ROBOTICA'2008*, Aveiro, Portugal, April 2008, pp. 27–32.

[14] F. Santos, G. Corrente, L. Almeida, N. Lau, and L. S. Lopes, "Self-configuration of an adaptive TDMA wireless communication protocol for teams of mobile robots," in *Proc. of the 13th Portuguese Conference on Artificial Intelligence, EPIA 2007*, GuimarÃ£es, Portugal, December 2007.

[15] A. J. R. Neves, G. Corrente, and A. J. Pinho, "An omnidirectional vision system for soccer robots," in *Progress in Artificial Intelligence*. 2007, vol. 4874 of *Lecture Notes in Artificial Inteligence*, pp. 499–507, Springer.

[16] A. J. R. Neves, D. A. Martins, and A. J. Pinho, "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 - ROBOTICA'2008*, Aveiro, Portugal, April 2008, pp. 51–55.

[17] D. A. Martins, A. J. R. Neves, and A. J. Pinho, "Real-time generic ball recognition in RoboCup domain," in *Proc. of the 11th edition of the Ibero-American Conference on Artificial Intelligence, IBERAMIA 2008*, Lisbon, Portugal, october 2008.

[18] J. E. Bresenham, "Algorithm for computer control of a digital plotter," *IBM Systems Journal*, vol. 4, no. 1, pp. 25–30, 1965.