Development of Local Vision-based Behaviors for a Robotic Soccer Player

Antonio Salim, Olac Fuentes, Angélica Muñoz

Reporte Técnico No. CCC-04-005
22 de Junio de 2004

Luis Enrique Erro 1
Sta. Ma. Tonantzintla,
72840, Puebla, México.

Development of Local Vision-based Behaviors for a Robotic Soccer Player

Antonio Salim Olac Fuentes Angélica Muñoz

Computer Science Department
National Institute of Astrophysics, Optics and Electronics
Luis Enrique Erro # 1, Santa María Tonantzintla, Puebla, 72840, México
E-mail: {asalimm,fuentes,munoz}@inaoep.mx

Abstract

This report describes the development of local vision-based behaviors for the robotic soccer domain. The behaviors, which include finding ball, approaching ball, finding goal, approaching goal, shooting *and* avoiding, *have been designed and implemented using a hierarchical control system. The* avoiding *behavior was learned using the C4.5 rule induction algorithm, the rest of the behaviors were programmed by hand. The object detection system is able to detect the objects of interest at a frame rate of 17 images per second. We compare three pixel classification techniques; one technique is based on linear color thresholds, another is based on logical AND operations and the last one is based on the artificial life paradigm. Experimental results obtained with a Pioneer 2-DX robot equipped with a single camera, playing on an enclosed soccer field with forward role indicate that the robot operates successfully, scoring goals in 90% of the trials.*

1 Introduction

Robotic soccer is a common task for artificial intelligence and robotics research [1], this task permits the evaluation of various theories, the design of algorithms and agent architectures. This report focuses on the design and evaluation of perceptual and behavioral control methods for the RoboCup Physical Agent Challenge [1], these methods are based on local perception, because it permits designers to program robust and reliable robotic soccer players, that are able to cope with highly dynamic environments such as RoboCup environments.

Vision is the primary sense used by robots in RoboCup. We used a local vision approach with an off-board computer. In this approach, the robot is equipped with a camera and an off-board image processing system determines the commands for the robot. We used this approach because of the advantages that it offers, which include lower power consumption, faster processing and the fact that inexpensive desktop computers can be used instead of specialized vision processing boards. We compared 3 strategies for pixel classification. One strategy was based on linear color thresholds, another was based on the algorithm of Bruce et al.[4] and the last one was based on the artificial life paradigm.

Behaviors were designed and implemented using a hierarchical control system with a memory module for a reactive robotic soccer player [2]. The behaviors, which include *finding ball, approaching ball, finding goal, approaching goal,* and *shooting*, were programmed by hand. The *avoiding behavior* was learned

via direct interaction with the environment with the help of a human operator using the C4.5 decision tree algorithm [3].

The report is organized as follows. Section 2 reviews related work. Section 3 describes the methodological approach used in the design of our robotic soccer player. Section 4 summarizes the experimental results obtained. Finally, Section 5 discusses conclusions and perspectives.

2 Related work

We survey a number of works in the field of vision and control for robotic soccer.

2.1 Vision

The cognachrome vision system©, manufactured by Newton Research Labs is a commercial hardware-based vision system used by several robot soccer teams [6]. Since it is hardware-based, it is faster than software running on a general purpose processor. Its disadvantages are its high cost and the fact that it only recognizes three different colors.

A number of past RoboCup teams have used alternative color spaces such as HSB or HSV proposed by Asada for color discrimination, since those separates color from brightness [7].

Several RoboCup soccer teams have adopted the use of omnidirectional vision generated by the use of a convex mirror [8]. This type of vision has the advantage of providing a panoramic view of the field, sacrificing image resolution. Moreover, the profiles of the mirrors are designed for a specific task.

The fast and cheap color image segmentation for interactive robots employs region segmentation by color classes [4]. This system has the advantage of being able to classify more than 32 colors using only two logical AND operations and it uses alternative color spaces.

For our vision system, we used the pixel classification technique proposed by Bruce [4] and a variant of the color spaces proposed by Asada [7] (for details see section 3.2).

2.2 Control

Takahashi et al. used multi-layered reinforcement learning which decompose a large state space at the bottom level into several subspaces and merges those subspaces at the higher level. Each module has its own goal state, and it learns to reach the goal maximizing the sum of the discounted reward received over time [10].

Steinbauer et al. used an abstract layer within their control architecture to provide the integration of domain knowledge such as rules, long term planning and strategic decisions. The origin of action planning was a knowledge base. This base contained explicit domain knowledge used by a planning module to find a sequence of actions that achieves a given goal [11]. The RMIT RoboCup team used a symbolic model of the world. The robot can use it to reason and take decisions [5].

Bonarini et al. developed reactive behaviors based on fuzzy logic. In this model, each behavior had associated two sets of fuzzy predicates representing its activating conditions and motivations. A distributed planner was used to weight the actions proposed by the behaviors [12].

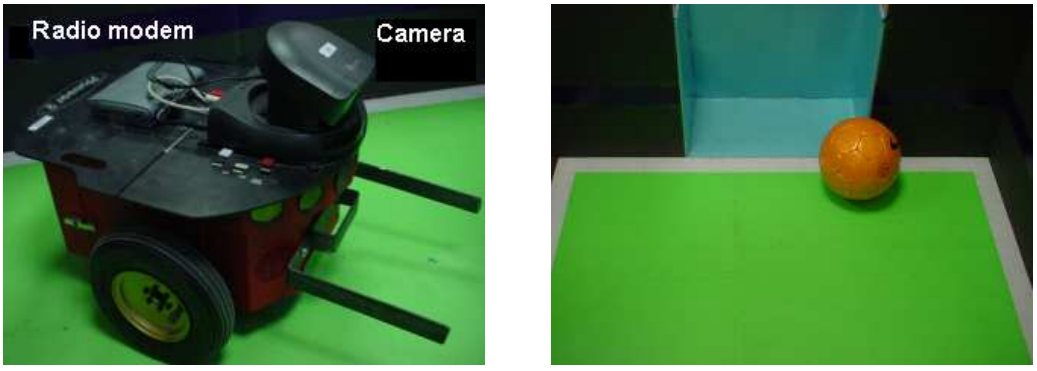
Gómez et al. used an architecture called dynamic schema hierarchies. In this architecture, the control and the perception are distributed on a schema collection structured in a hierarchy. Perceptual schemas produce information that can be read by motor schemas to generate their outputs [13].

We used a behavior-based control system or subsumption architecture with a memory module in order to control our robotic soccer player (for details see section 3.3).

3 The System

3.1 Hardware and settings

The robot used in this research is a Pioneer 2-DX mobile robot made by Activ-Media©, equipped with a Pioneer PTZ camera, a manually-adapted fixed gripper and a radio modem. The dimensions of the robot are 44 cm long, 38 cm wide and 34 cm tall, including the video-camera. The robot is remotely controlled by a AMD Athlon 1900 computer with 512 MB of RAM. Figure 1(a) shows a picture of our robotic soccer player.



(a) (b)

Figure 1. The robotic soccer player (a). The soccer playing field (b).

The environment for the robot is an enclosed playing field with a size of 180 cm in length and 120 cm in width. There was only one goal, painted cyan, centered in one end of the field with a size of 60 cm wide and 50 cm tall. The walls were marked with an auxiliary purple line whose height is 20 cm from the floor. Figure 1(b) shows a picture of the playing field.

3.2 Vision

A robust, fast and fault tolerant vision system is fundamental for the robot, since it is the only source of information about the state of the environment. Since all objects of interest in the environment are colored, we believe that vision is the most appropriate sensor for a robot that has to play soccer. We present below the object detection system used by the robot and a strategy for pixel classification based on the artificial life paradigm.

3.2.1 Object detection

The vision system processes images captured by the robot's camera and reports the locations of various objects of interest relative to the robot's current location. The objects of interest are the orange ball, the cyan goal and the auxiliary purple line on the field's wall. The steps of our object detection method are:

1. *Image capture:* Images are captured in RGB in a 160×120 resolution.
2. *Image resizing:* The images are resized to 80×60 pixels.

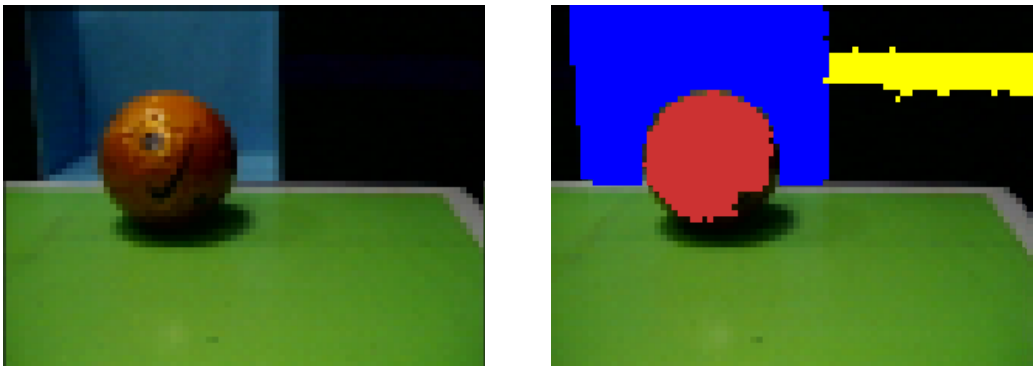
3. *Color space transformation:* The RGB images are transformed into the HUV color space.

4. *Pixel classification:* Each pixel is classified by predetermined color thresholds in RGB and HUV color spaces. There are 3 color classes: the colors of the ball, the goal, and the auxiliary line. The pixel classification is based on [4] in order to use only two logical AND operations for each color space.

5. *Region segmentation:* Pixels of each color class are grouped together into connected regions.

6. *Object filtering:* False positives are filtered out via region size.

Figure 2(a) shows an image captured by the frame grabber and Figure 2(b) shows the robot's perception.



(a) (b)

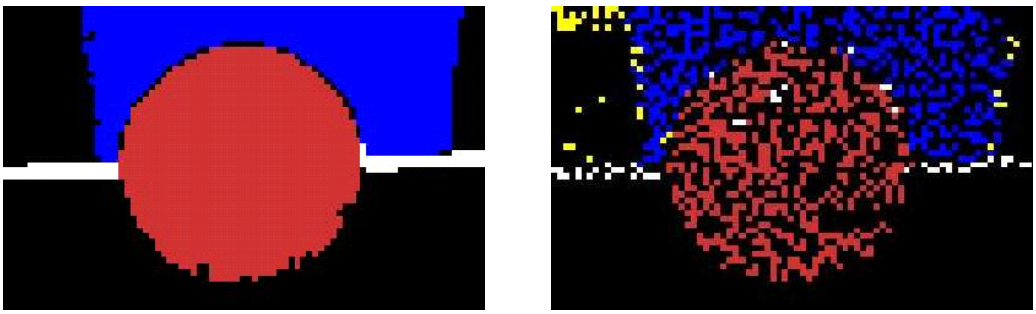
Figure 2. Image captured by the camera (a). The robot's perception (b).

3.2.2 Artificial life approach for pixel classification

In order to reduce the time invested in pixel classification, the most expensive step in object detection, we tested an artificial life-based method. Ideas of distributed computing were taken from Reynolds's boids [9], where a group of agents moves as a flock of birds or a school of fish. For this strategy, we used 2500 agents, each having an internal state to indicate if it is over an object of interest or not. Agents were able to detect 3 color classes: the colors of the ball, the goal and the auxiliary line in the walls. Agents were serialized by an agent manager which assigned movement turns and prevented collisions between agents. However, the recognition task is distributed among agents. The agents can move in their world which is the image perceived by the camera. Only one agent can be situated over each pixel. Agents can sense the color intensity values in the image in order to perform pixel classification. The locomotion of an agent consists of moving pixel by pixel via its actuators. Figure 3 shows a snapshot of the pixel classification methods.

3.3 Control

Behaviors were designed and implemented using a subsumption architecture [2] because this architecture offers the necessary reactivity for dynamic environments. We incorporated a new element to this architecture, a memory module. This module acts as a short-term memory that enables the robot to remember past events that can be useful for future decisions. The memory module affects directly the behaviors programmed into the robot.



(a) (b)

Figure 3. Bruce-based pixel classification (a). Artificial life-based pixel classification (b).

The *avoiding* behavior is a horizontal behavior in the architecture that overwrites the output of the rest of the behaviors in our vertical subsumption architecture. The architecture was implemented using four threads in C++, one for the vertical behaviors module, one for the memory module, one for controlling the robot movements and one for the horizontal behavior to avoid collisions with the walls. In this architecture, each behavior has its own perceptual gathering, which is responsible of sensing the objects of interest. Each behavior writes its movement commands to shared memory to be executed. The architecture used for the robot's control system is shown in Figure 4.

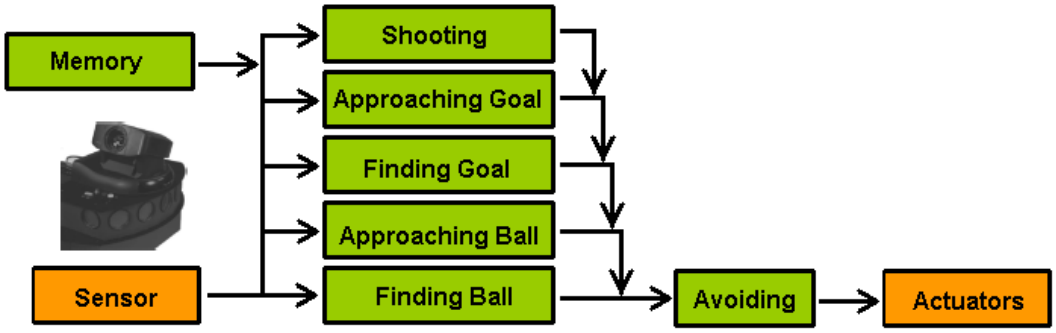


Figure 4. The architecture of the system.

3.4 Description of modules and behaviors

1. *Memory:* This is an essential module for the achievement of the robot's global behavior. Memory, like behaviors, has its own perceptual gathering to sense the ball and the goal. The function of this memory is to remember the last direction in which the ball or the goal were perceived with respect to the point of view of the robot. The memory module affects directly the other behaviors because it writes the directions of the ball and the goal on a shared memory used in the behaviors's execution. There are 6 possible directions that the memory has to remember: ball to the left, ball to the right, centered ball, goal to the left, goal to the right and centered goal.

2. *Finding ball:* The robot executes a turn over its rotational axis until the ball is perceived. The robot turns in the direction in which the ball was last perceived. If this information was not registered then

the robot executes a random turn towards the left or right.

3. *Approaching ball:* The robot centers and approaches the ball until the ball is at an approximate distance of 1 cm.

4. *Finding goal:* The robot executes a turn over its rotational axis until the goal is perceived. The robot turns in the direction in which the goal was last perceived. If this information was not registered then the robot executes a random turn towards the left or right.

5. *Approaching goal:* The robot executes a turn in the direction of the center of the goal until the goal is centered with respect to the point of view of the robot.

6. *Shooting:* The robot makes an abrupt increase of its velocity to shot the ball towards the goal. There are two possible kind of shots, a short shot when the robot is close to the goal (a distance equal or less than 65 cm) and a long shot, when the robot is far from the goal (more than 65 cm).

7. *Avoiding:* The robot avoids crashing against the walls that surround the soccer field. Determining manually the necessary conditions in which the robot collides with the wall is difficult because the wall can be perceived in many forms, therefore we used a machine learning technique called C4.5 [3] to learn whether a collision must be avoided or not. With the help of a human operator, the robot was situated in 153 cases where there is a collision and 293 cases where there is no collision. We use 10-fold cross-validation and selected the best decision tree, with 92.37% of classification accuracy. Finally, the rules obtained in the training phase were implemented in the *avoiding* behavior, these rules are shown in Figure 5.

The global behavior of our robotic soccer player is described by the automaton in Figure 6.

4 Experimental results

4.1 Pixel classification results

We present the results obtained by three implementations of pixel classification. The first implementation was performed based on linear color thresholds, the second implementation was based on the algorithm proposed by Bruce et al. for pixel classification [4], and finally, the third implementation was based on the artificial life paradigm.

Method	Images per second	Processing average time
Linear color thresholds	12 images	0.0874 sec.
Bruce-based method	18 images	0.0553 sec.
Artificial life-based method	14 images	0.0707 sec.

Table 1. Pixel classification Results.

Results of pixel classification are shown in Table 1. As this table indicates, the worst of the strategies for pixel classification task was based on linear color thresholds. The best strategy for this task was based on the algorithm proposed by Bruce et al. [4], this strategy was implemented as a step in the object detection system for the robotic soccer player. We expected a better performance from the pixel classification method based on artificial life, because this method needs to examine only 2500 pixels, corresponding to the total

```
J48 pruned tree
------------------

dsrRow21 <= 31
|   dPhi1 <= -0.195257
|   |   dsrColumn21 <= 42: 0 (2.0)
|   |   dsrColumn21 > 42
|   |   |   dPhi <= 2.920391: 1 (12.0)
|   |   |   dPhi > 2.920391: 0 (3.0/1.0)
|   dPhi1 > -0.195257
|   |   dContLength <= 208.870056
|   |   |   dMeanLength <= 12.299213
|   |   |   |   dicRow <= 10: 0 (23.0/1.0)
|   |   |   |   dicRow > 10: 1 (2.0)
|   |   |   dMeanLength > 12.299213: 0 (242.0)
|   |   dContLength > 208.870056
|   |   |   dConvexity <= 0.698842: 1 (4.0)
|   |   |   dConvexity > 0.698842
|   |   |   |   dCompactness <= 2.466859: 1 (4.0/1.0)
|   |   |   |   dCompactness > 2.466859: 0 (10.0)
dsrRow21 > 31
|   dsrRow21 <= 34
|   |   dRoundness <= 0.53691: 1 (43.0/3.0)
|   |   dRoundness > 0.53691
|   |   |   dContLength <= 224.225403
|   |   |   |   dicColumn <= 17: 0 (2.0)
|   |   |   |   dicColumn > 17: 1 (9.0/1.0)
|   |   |   dContLength > 224.225403: 0 (7.0)
|   dsrRow21 > 34: 1 (83.0/1.0)

Number of Leaves  :     14

Size of the tree :      27
```

Figure 5. Rules obtained for the avoiding behavior. Class 1 indicates collision and class 0 indicates no collision.

number of agents, instead of the total number of pixels in the image (8600 pixels). However, in this strategy each of the agents spends time calculating its next movement, producing a general medium performance.

4.2 Avoiding behavior results

For the *avoiding* behavior, we collected a training set of 446 instances of collisions. There were 153 positive samples where there was a collision and 293 negative samples where there was not collision. The experiments were validated using 10-fold cross-validation. We tested 5 machine learning algorithms for the classification task, the results obtained are summarized in Table 2. This table shows the classification results obtained for each machine learning algorithm. As the results show, the C4.5 algorithm obtained the best percentage of correctly classified instances for the collision avoidance task. The rules generated by C4.5 algorithm were implemented in our *avoiding* behavior.

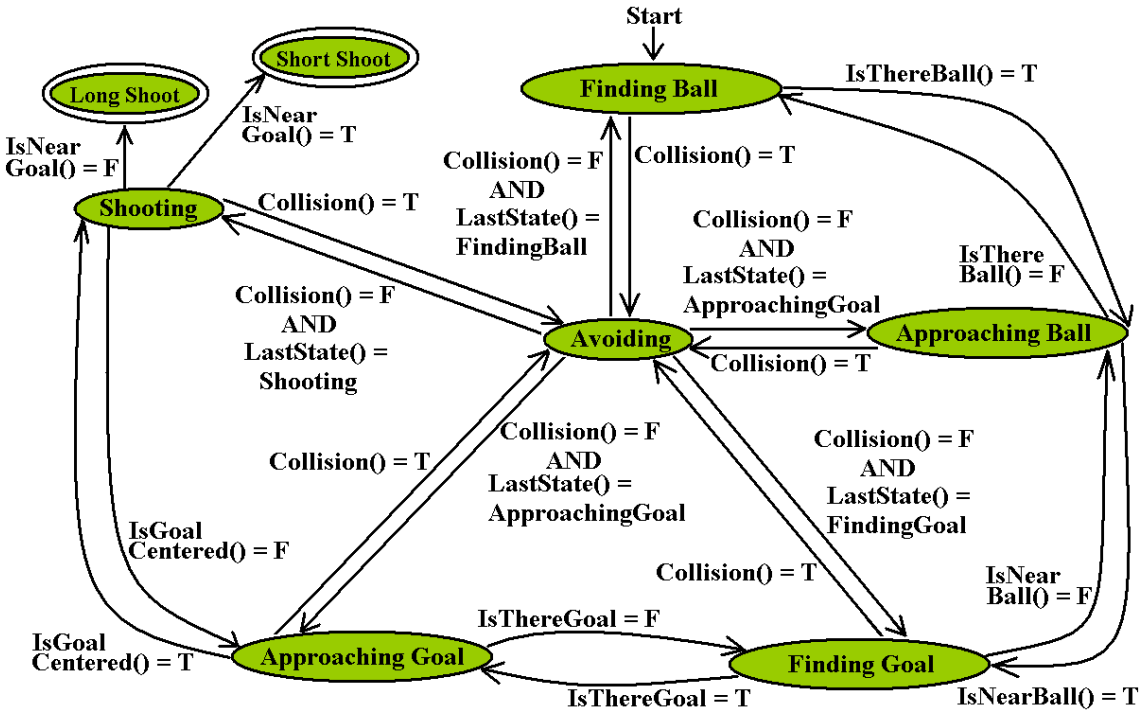


Figure 6. Automaton resuming the global behavior of the robot. IsThereBall(), IsNearBall(), IsThereGoal(), IsGoalCentered(), IsNearGoal() and Collision() are boolean functions to indicate: if the ball is percibed, if the ball is near to the robot, if the goal is percibed, if the goal is centered with respect to the point of view of the robot, if the goal is near to the robot and if a collision with the walls is detected, respectively. The values of return for the boolean functions are: T (true) or F (false). The function LastState() returns the last state visited in the automaton.

4.3 Global performance

Our robotic soccer player has a forward role, thus its main task is to score goals in a minimum amount of time. In order to test the global performance of our robotic soccer player, we designed a set of experiments. The experiments were performed on the soccer field shown in Figure 1(b). The robot position, robot orientation and ball position were selected 20 times randomly as follows:

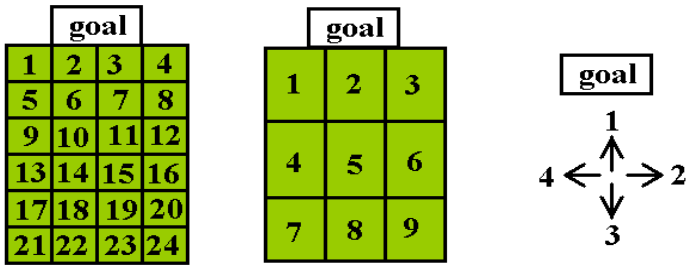
1. For selecting the robot position, the field was divided into 24 cells of equal size. Figure 7(a) shows the cells for the robot's position.

2. For selecting the ball position, the field was divided into 9 cells of equal size. Figure 7(b) shows the cells for the ball's position.

3. For selecting the robot orientation, there were 4 directions to the robot. The orientation where the goal is: $1)$ in front of the robot, $2)$ left to the robot, $3)$ back to the robot and $4)$ right to the robot. Figure 7(c) shows the possible orientations for the robot.

Machine learning algorithm	% of correctly classified instances
Support Vector Machines	91.25 ± 0.0603 %
Artificial Neural Networks	90.40 ± 0.0849 %
C4.5	92.20 ± 0.0638 %
Naive Bayes	87.62 ± 0.0683 %
Conjuntive Rules	90.68 ± 0.0222 %

Table 2. Percentage of correctly classified instances by machine learning algorithm for the *avoiding* behavior.



(a) (b) (c)

Figure 7. Experiments's configuration. Robot position (a). Ball position (b). Robot orientation (c).

An experiment's configuration can be represented as a triplet, of the form *(ball position, robot position, robot orientation)*. The configuration for the 20 experiments performed were: *(24,7,1), (24,8,1), (21,2,2), (8,8,4), (18,7,3), (22,9,2), (24,4,4), (7,4,3) , (6,4,3), (8,2,2), (15,1,3), (21,4,1), (12,2,2), (11,9,1), (7,8,4), (20,9,1), (7,9,4), (11,9,4), (10,5,2)* and *(6,2,3)*.

Table 3 summarizes the time spent in seconds by each behavior performed by the robot in the experiments. The total time spent for the robot in the experiments was 632 seconds.

The percentage of time used by behaviors in the experiments was 28% for *Finding Ball*, 32.27% for *Approaching Ball*, 14.24% for *Finding Goal*, 9.49% for *Approaching Goal* and 16% for *Shooting*. As these results indicate, the robot spent most of its time executing the behavior *approaching ball*. The *avoiding* behavior was successful, the robot avoided 10 of 12 avoidance situations (83% success). The average time required by the robot to score a goal is 35.11 seconds.

An useful functionality of the soccer player emerges from the interaction of 3 behaviors: *approaching ball*, *finding goal* and *avoiding*. This emergent behavior consist of *regaining the ball from the corner*. In the experiments the robot was able to regain the ball from the corner in four out of five times (80% success). In the 20 experiments executed, the robot was able to score 18 goals (90% success).

5 Conclusions

In this report we presented our research on the development of local vision-based behaviors for a Pioneer 2-DX robot equipped with a single camera.

The subsumption architecture used for the robot control gives the necessary reactivity to play soccer.

Experiment Number	Finding Ball	Approaching Ball	Finding Goal	Approaching Goal	Shooting	Duration
1	16	10	10	–	6	42
2	11	19	17	1	6	54
3	–	–	–	–	–	–
4	13	9	3	4	5	34
5	8	5	–	2	6	21
6	8	11	7	10	6	42
7	6	31	6	3	6	52
8	5	9	7	1	5	27
9	5	6	–	5	5	21
10	8	6	–	4	5	23
11	2	16	8	–	6	32
12	17	20	11	6	6	60
13	–	–	–	–	–	–
14	–	7	–	3	6	16
15	16	11	–	7	5	39
16	–	5	–	–	6	11
17	27	8	–	4	6	45
18	19	8	–	3	6	36
19	6	12	11	2	5	36
20	10	11	10	5	5	41
Totals	177	204	90	60	101	632 sec

Table 3. Spending time in seconds in the behaviors executed by the robot during 20 experiments.

Even though the robot displays a highly reactive behavior, the memory that we incorporated enables the robot to base its decisions on past events. The *avoidance* behavior was much easier to learn than to program by hand. Building the avoiding behavior using the C4.5 algorithm to learn to avoid collisions with the walls was successful.

Although the strategy for pixel classification based on artificial life did not improve the performance, it seems to be a promising strategy to create a completely distributed control system for a robotic soccer player. The main limitation of this approach is the current computational processing power to support a large number of agents with complex behaviors.

Using our object detection method we can detect the ball, goal and auxiliary line, at a frame rate of 17 frames per second.

The *avoidance* behavior was much easier to learn than to program by hand. Building the avoiding behavior using the C4.5 algorithm to learn to avoid collisions with the walls was successful.

In future work, we will use other machine learning techniques such as artificial neural networks or support vector machines to help us develop behaviors such as *approaching ball*.

References

[1] Asada, M., Stone, P., Kitano, H., Werger, B., Kuniyoshi, Y., Drogoul, A., Duhaut, D., Veloso, M.: The RoboCup Physical Agent Challenge: Phase-I. Applied Artificial Intelligence. 12 (1998) 251–263.

[2] Brooks, R. A.: A Robust Layered Control System for a Mobile Robot. IEEE Journal of Robotics and Automation. RA-2 (1986) 14–23.

[3] Quinlan, J. R.: C4.5: Programs for Machine Learning. Morgan Kaufmann, San Mateo, C.A. (1993).

[4] Bruce, J., Balch, T., Veloso, M.: Fast and inexpensive color image segmentation for interactive robots. In Proc. of the 2000 IEEE/RSJ International Conference on Intelligent Robots and Systems. (2000) 2061–2066

[5] Brusey, J., Jennings, A., Makies, M., Keen, C., Kendall, A., Padgham, L., Singh, D., Plöger, P., Schöll, P., Siegberg, A., Streit, A., Verbeek, C., Wilberg, J.: Team RMIT. RoboCup-99 Team Descriptions Middle Robots League, Team GMD Robots. (1999) 181–188.

[6] Werger, B. B., Funes, P., Schneider-Fontán, M., Sargent, R., Witty, C., Witty, T.: The Spirit of Bolivia: Complex behavior through minimal control. Lecture Notes in Computer Science, Springer-Verlag. 1395 (1997) 348–356.

[7] Asada, M., Kitano, H.: RoboCup-98: Robot Soccer World Cup II. Lecture Notes in Computer Science, Springer-Verlag. 1604 (1999).

[8] Bonarini, A., Aliverti, P., Lucioni, M.: An omnidirectional vision sensor for fast tracking for mobile robots. Transactions on Instrumentation and Measurement. 49 (2000) 509–512.

[9] Reynolds, C. W.: Flocks, Herds, and Schools: A Distributed Behavioral Model. Computer Graphics. 21 (1987) 25–34.

[10] Takahashi, Y., Asada, M.: Vision-guided behavior acquisition of a mobile robot by multi-layered reinforcement learning. IEEE/RSJ International Conference on Intelligent Robots and Systems. 1 (2000) 395–402.

[11] Steinbauer, G., Faschinger, M.: The Mostly Harmless RoboCup Middle Size Team. OGAI. 22 (2003) 8–13.

[12] Bonarini, A., Invernizzi, G., Halva, T, Matteucci, M.: A fuzzy architecture to coordinate robot behaviors. Fuzzy Sets and Systems. 134 (2003) 101–115.

[13] Gómez, V. M., Cañas, J. M., San Martín, F., Mantellán, V.: Vision-based schemas for an autonomous robotic soccer player. Proceedings of IV WAF'2003. (2003) 109–120.