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

Antonio Salim Olac Fuentes Angélica Muñoz

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

Abstract

This research focuses on 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 robust layered control system. The avoiding behavior was learned using the C4.5 decision tree algorithm, the rest of the behaviors were programmed by hand. We describe the vision system employed by a mobile robot. Additionally, we compare two pixel classification techniques. One technique is based on the fast and cheap color image segmentation for interactive robots and the other one is based on the artificial life paradigm. We describe experimental results obtained using a Pioneer 2-DX robot equipped with a single camera, playing on an enclosed soccer field with forward role.

1. Introduction

Robotic soccer is a common task for artificial intelligence and robotics research [3, 12]. This task provides a good test bed for evaluation of various theories, algorithms and agent architectures. This research focuses on designing and evaluating perceptual and behavioral control methods for the RoboCup Physical Agent Challenge [3].

Vision is the primary sense used by robots in RoboCup. When designing the robots, researchers have two different types of vision systems available: global vision and local vision. In global vision, a camera is mounted over the field. The image captured by the camera is passed to an external computer that processes the image and determines the commands for the robot. In local vision, the robot is equipped with a camera and the image processing system on-board or off-board determines the commands for the robot. Our primary goal in this research is the development of local vision-based behaviors for a Pioneer 2-DX robot equipped with a single camera. We used a local vision approach with an off-board computer because of the advantages that this method offers, which include lower power consumption, faster processing and inexpensive desktop computers instead of specialized vision processing boards. We compare two strategies for pixel classification. One strategy is based on the fast and cheap color image segmentation for interactive robots[8] and the other one is based on the artificial life paradigm.

Behaviors were designed and implemented using a robust layered control system with a memory module for a reactive robotic soccer player [7]. 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 [13].

The paper 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 sumarizes the experimental results obtained. Finally, section 5 discusses conclusions and perspectives. [1]

2. Related Work

Designing a robot to play soccer is very challenging because the robot should incorporate the design principles of autonomous robots, multi-agent collaboration, strategy acquisition, real-time reasoning, strategic decision making, intelligent robot control, and machine learning. We survey a number of works in the field of vision and control for robotic soccer.

2.1. Vision

Template matching was used by Cheng and Zelinsky in the vision system for their autonomous soccer robots [10].

[1]Accepted to Encuentro Internacional de Ciencias de la Computación'04

In template matching, objects can be identified by comparison with stored object templates against the perceived image. Template matching can fail if the intensity varies significantly over areas where the template is applied.

The cognachrome vision system [1], manufactured by Newton Research Labs is a commercial hardware-based vision system used by several robot soccer teams [1, 17]. Since it is hardware-based, is faster than software running on a general purpose processor. Its disadvantages are its high cost (2450 dollars) and that it only recognizes 3 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 it separates color from brightness [2].

Several RoboCup soccer teams have adopted the use of omnidirectional vision generated by the use of a mirror [4]. This type of vision has the advantage of getting a panoramic view of the field but it is often necessary to correct the distortion generated by the mirror.

The fast and cheap color image segmentation for interactive robots employs region segmentation by color classes [8]. This system has the advantage of being able to classify more than 32 colors using only two logical AND operations and it can uses alternative color spaces.

For our vision system, we used the pixel classification proposed by Bruce [8] and a variant of the color spaces proposed by Asada [2] (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 [16].

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 [15]. The RMIT RoboCup team used a symbolic model of the world. The robot can use it to reason and take decisions [9].

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 [5].

Bredenfeld et al. used the ”dual dynamics” model of behavior control. This is a mathematical model of a control system based on behaviors. The robot’s behaviors are specified through differential equations [6].

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 [11].

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 3.5).

3. Proposal

3.1. Hardware and Settings

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

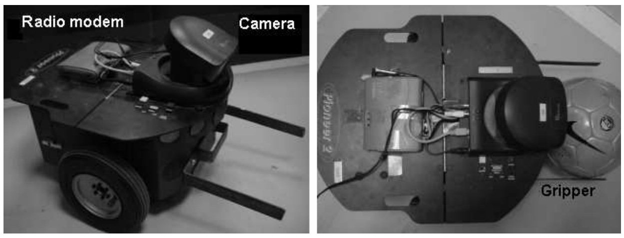

Figure 1. The robotic soccer player. A lateral view (left) and a superior view (right).

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 an extreme 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 2 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. Because of all objects of interest in the environment are colored, we believe that vision is the most appropriate sensor for a robot that has

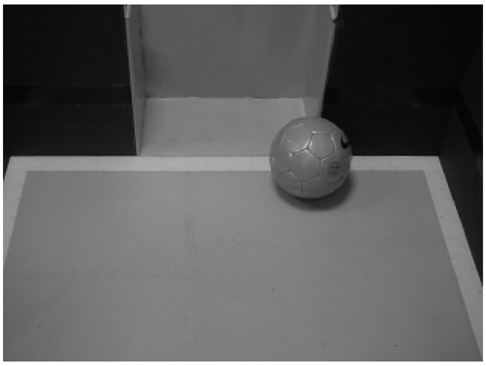

Figure 2. The soccer playing field.

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.

3.3. 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 in the field's wall. The steps of our object detection method are:

1. *Image capture:* Images are captured in RGB in resolution 160×120 pixels.
2. *Image resizing:* The images captured 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 [8] 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.

We are using the smallest resolution (80×60 pixels) for the images where an object of interest is distinguished by its color, size and form. We consider that images with higher resolution are not necessary for object detection. The color space transformation is a necessary step because we can classify pixels with a minimum and maximum threshold and the light sensitivity is reduced. The region segmentation step is used to consider groups of pixels that have more probability to be part of an object of interest instead of isolated pixels that can be noise. The object filtering step discards small segments of grouped pixels because they can be noise in the image. If the number of pixels that form a region is bigger than a preestablished threshold then the region is considered as an object of interest. The final result of the color classification is a new image indicating the color class membership for each pixel. Figure 3 on the left shows an image captured by the frame grabber and on the right shows the robot's perception.

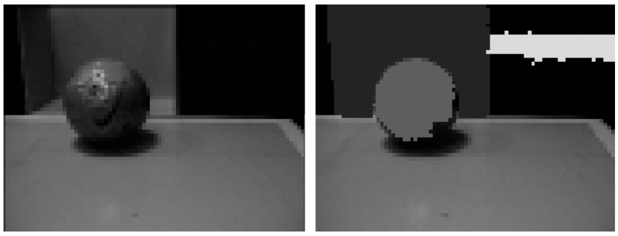

Figure 3. Image captured by the camera (left) and the robot's perception (right).

3.4. Artificial life approach for pixel classification

In order to reduce the time invested in pixel classification, the most consuming step in object detection, we tested an artificial life-based method. Ideas of distributed computing were taken from Reynolds's boids [14] where a group of agents moves as a flock of birds or a school of fish. For this strategy, we used 2500 agents which had 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 controlled by an agent manager which gave movement turns and prevented collisions between agents. The agents can move in their world that is the image perceived by the camera. Only one agent can be situated over a pixel. Agents can sense the gray values in the image in order to perform pixel classification. The locomotion of an agent consists moving pixel by pixel via its actuators. Figure 4 shows a snapshot of the pixel classification methods.

3.5. Control

Behaviors were designed and implemented using a robust layered control system or subsumption architecture [7] because this architecture offers the necessary reactivity for dynamical environments. We incorporated a new element to this architecture, a memory module. This module acts as a

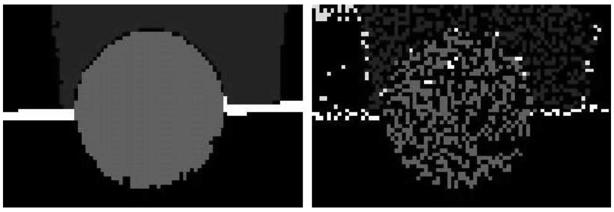

Figure 4. Pixel classification snapshots with both approaches. Bruce-based (left) and artificial life-based (right).

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.

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 a shared memory to be executed. The architecture used for the robot's control system is shown in Figure 5.

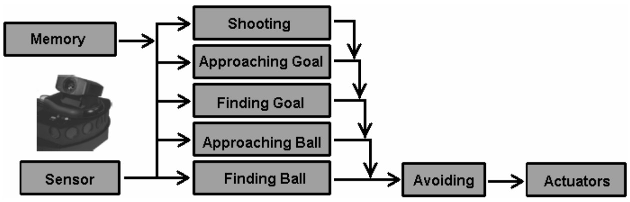

Figure 5. The architecture of the system.

3.6. Description of modules and behaviors

1. *Memory:* The memory is not a formal behavior in the architecture, but is an essential module for the achievement of the robot's global behavior. Memory like behaviors has their 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 towards the direction in which the ball was perceived last time. 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 towards the direction in which the goal was perceived last time. 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 towards 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 shoot the ball towards the goal. There are two possible kind of shoots, a short shoot when the robot is close to the goal (equal or less than 65 cm) and a long shoot when the robot is far from the goal (more than 65 cm).
7. *Avoiding:* The robot avoids to hit 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 to learn whether a collision must be avoid or not [13]. 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 a 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 6.

The global behavior of our robotic soccer player is described by the automaton in Figure 7.

4. Experimental results

4.1. Pixel classification results

We present the results obtained by three implementations of pixel classification. The descriptions of the pixel classi-

```
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 6. Rules obtained for the avoiding behavior. Class 1 indicates collision and class 0 indicates no collision.

fication experiments were:

1. When the pixel classification strategy was implemented using linear color thresholds. The linear color thresholds partitioning the color space with linear boundaries (e.g. planes in 3-dimensional spaces). A particular pixel is then classified according to which partition it lies in.

2. When the pixel classification strategy was based on the fast and cheap color image segmentation for interactive robots or Bruce's work [8].

3. When the pixel classification strategy was based on artificial life paradigm.

Results of pixel classification are shown in Table 1. As this table indicates, the worst of the strategies was the first

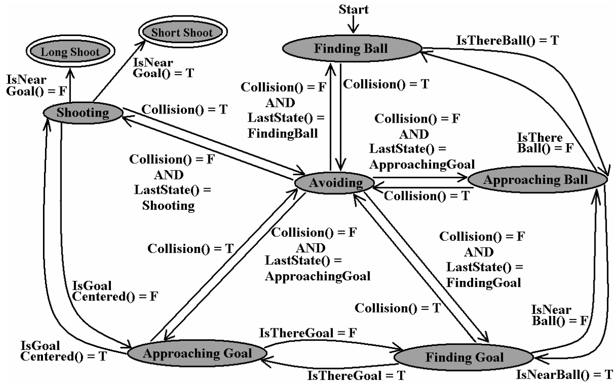

Figure 7. Automaton resuming the global behavior of the robot. IsThereBall(), IsNearBall(), IsThereGoal(), IsGoalCentered(), IsNearGoal() and Collision() are boolean functions to indicate: if the ball is percibed, if the ball is near to the robot, if the goal is percibed, if the goal is centered with 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.

and the best strategy was selected for our robotic soccer player as part of the object detection system. We expected a better performance of the pixel classification based on artificial life, because this method needs to examine only 2500 pixels, the total 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 labelled with class 1 and 293 negative samples labelled with class 0. 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.

#	Images per second	Processing average time
1	12 images	0.0874 sec.
2	18 images	0.0553 sec.
3	14 images	0.0707 sec.

Table 1. Pixel classification Results. 1) Pixel classification using logical and relational comparisons, 2) Pixel classification based on the work of Bruce and 3) Pixel classification based on artificial life paradigm.

#	Algorithm	% of correctly classified instances
1	Support Vector Machines	91.25%
2	Artificial Neural Networks	90.13%
3	C4.5	92.37%
4	Naive Bayes	87.87%
5	Conjuntive Rules	90.80%

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

4.3. Global performance

Our robotic soccer player has a forward role, then 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 realized in the soccer field showed in Figure 2. The robot's position, robot's direction and ball's position were selected 5 times randomly as follows:

1. For selecting the robot's position, the field was divided into 24 cells of equal size (Figure 8 a) shows the cells for the robot's position).

2. For selecting the ball's position, the field was divided into 9 cells of equal size (Figure 8 b) shows the cells for the ball's position).

3. For selecting the robot's direction, there were 4 directions to the robot. The direction 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 8 c) shows the possible directions for the robot).

An experiment's configuration can be represented as a tuple of 3 elements, of this form (ball's position, robot's position, robot's direction). The configuration for the 5 experiments performed were: (4,6,4), (11,1,2), (13,8,4), (9,7,1) and (20,7,2). Table 3 shows the typical sequence of behaviors to score a goal. The steps show in this table were performed by the robot in experiment 1 (configuration (4,6,4)).

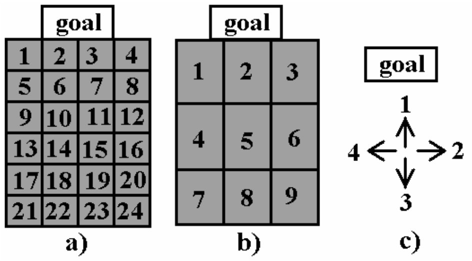

Figure 8. Experiments's configuration. a) Robot's position. b) Ball's position. c)Robot's direction.

#	Behavior	Start	End	Total Duration
1	FB	0 sec	13 sec	13
2	AB	13 sec	20 sec	7
3	FG	20 sec	22 sec	2
4	AG	22 sec	24 sec	2
5	AB	24 sec	25 sec	1
6	AG	25 sec	29 sec	4
7	S	29 sec	35 sec	6
				35 sec

Table 3. Typical sequence of behavior to score a goal. FB is Finding Ball, AB is Approaching Ball, FG is Finding Goal, AG is Approaching Goal, S is Shoot. Start, End and Duration are indicated in seconds.

Table 4 summarizes the spent time in seconds by each behavior performed by the robot in the 5 experiments. The total time for the experiments was 208 seconds.

The percentage of time used by behaviors in the experiments was 25.48% for Finding Ball, 27.88% for Approaching Ball, 21.15% for Finding Goal, 12.5% for Approaching Goal and 12.98% for Shooting. As these results indicate, the robot spent the most of its time executing the behavior approaching ball. The average time required by the robot to score a goal is 41.6 seconds. In the 5 experiments executed, the robot was always able to score a goal.

5. Conclusions

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

#	FB	AB	FG	AG	S	Duration
1	13	8	2	6	6	35
2	16	16	25	6	5	68
3	12	8	6	2	5	33
4	–	6	–	5	5	16
5	12	20	11	7	6	56
	53	58	44	26	27	208 sec

Table 4. Spending time in seconds in the behaviors executed by the robot during 5 experiments. FB is Finding Ball, AB is Approaching Ball, FG is Finding Goal, AG is Approaching Goal, S is Shoot. Duration is indicated in seconds.

The subsumption architecture used for the robot control gives the necessary reactivity to play soccer. Even though the robot displays a highly reactive behavior, the memory that we incorporated enables the robot to base its decisions on past events.

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, in 17 images per second. The most consuming step for object detection is the image capture step. The main delay is produced by the frame grabber.

The avoidance behavior was much simpler to learn than to be programmed by hand. Building the avoiding behavior using 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 to develop behaviors such as *approaching ball*.

Finally, our robotic soccer robot increases currently its velocity to shoot the ball towards the goal. We consider to implement in the near future a kicking device to improve the shooting behavior.

References

[1] Newton labs inc. the cognachrome vision system.

[2] M. Asada and H. Kitano, editors. *RoboCup-98: Robot Soccer World Cup II*, volume 1604 of *Lecture Notes in Computer Science*. Springer-Verlag, 1999.

[3] M. Asada, P. Stone, H. Kitano, B. Werger, Y. Kuniyoshi, A. Drogoul, D. Duhaut, and M.Veloso. The robocup physical agent challenge: Phase-i. *Applied Artificial Intelligence*, 12(2-3), 1998.

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

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

[6] A. Bredenfeld, T. Christaller, W. Göhring, H. Günter, H. Jaeger, H. Kobialka, P. Plöger, P. Schöll, A. Siegberg, A. Streit, C. Verbeek, and J. Wilberg. Behavior engineering with "dual dynamics" models and design tools. *RoboCup-99 Team Descriptions Middle Robots League Team GMD Robots*, pages 134–143, 1999.

[7] R. A. Brooks. A robust layered control system for a mobile robot. *IEEE Journal of Robotics and Automation*, (RA-2), April 1986.

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

[9] J. Brusey, A. Jennings, M. Makies, C. Keen, A. Kendall, L. Padgham, and D. Singh. Rmit robocup team. *RoboCup-99 Team Descriptions Middle Robots League Team RMIT*, pages 181–188, 1999.

[10] G. Cheng and A. Zelinsky. Real-time vision processing for a soccer playing mobile robot. *Lecture Notes in Computer Science*, 1395:144–155, 1997.

[11] V. M. Gómez, J. M. Cañas, F. S. Martín, and V. Mantellán. Vision-based schemas for an autonomous robotic soccer player. *Proceedings of IV Workshop de Agentes Fsicos WAF-2003*, pages 109–120, 2003.

[12] H. Kitano, M. Tambe, P. Stone, M. Veloso, S. Coradeschi, E. Osawa, H. Matsubara, I. Noda, and M. Asada. The robocup synthetic agent challenge 97. *In Proc. of the Fifteenth International Joint Conference on Artificial Intelligence*, pages 24–29, 1997.

[13] J. R. Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufmann, 1993.

[14] C. W. Reynolds. Flocks, herds, and schools: A distributed behavioral model. *Computer Graphics*, 21:25–34, July 1987.

[15] G. Steinbauer and M. Faschinger. The mostly harmless robocup middle size team. *OGAI*, 22:8–13, 2003.

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

[17] B. B. Werger, P. Funes, M. Schneider-Fontán, R. Sargent, C. Witty, and T. Witty. The spirit of Bolivia: Complex behavior through minimal control. *Lecture Notes in Computer Science*, 1395:348–356, 1997.

[18] F. Young, B. Ng, K. Loh, E. Ong, K. Teo, and K. Loh. Alpha++. *RoboCup-99 Team Descriptions Middle Robots League Team BengKiat*, pages 107–112, 1999.