Multi-Fidelity Robotic Behaviors: Acting With Variable State Information

Elly Winner and Manuela Veloso
Computer Science Department
Carnegie Mellon University
Pittsburgh, PA 15213
{elly,veloso}@cs.cmu.edu
http://www.cs.cmu.edu/{~elly,~mmv}

Abstract

Our work is driven by one of the core purposes of artificial intelligence: to develop real robotic agents that achieve complex high-level goals in real-time environments. Robotic behaviors select actions as a function of the state of the robot and of the world. Designing robust and appropriate robotic behaviors is a difficult because of noise, uncertainty and the cost of acquiring the necessary state information. We addressed this challenge within the concrete domain of robotic soccer with fully autonomous legged robots provided by Sony. In this paper, we present one of the outcomes of this research: the introduction of *multi-fidelity behaviors* to explicitly adapt to different levels of state information accuracy. The paper motivates and introduces our general approach and then reports on our concrete work with the Sony robots. The multi-fidelity behaviors we developed allow the robots to successfully achieve their goals in a dynamic and adversarial environment. A robot acts according to a set of behaviors that aggressively balance the cost of acquiring state information with the value of that information to the robot's ability to achieve its high-level goals. The paper includes empirical experiments which support our method of balancing the cost and benefit of the incrementally-accurate state information.

Introduction

Intelligent agents operating in dynamic domains rely heavily on real-time information about the world around them to direct and control their behaviors. This information may be raw sensor data, processed sensor data, or sensor data that the agent has to spend effort to acquire.

In most realistic domains, raw sensor data is not refined enough to allow for high-level deliberation or control. The agent must spend effort to process the data it receives from its sensors to support the internal state representation necessary for higher-level control. In addition, sometimes the sensor data the agent needs to select behaviors is not immediately available. The agent must spend time to acquire the raw data in addition to the time spent processing it.

Thus, in many complex and dynamic domains, the quality of the agent's information about the world around it is dependent on the amount of resources the agent is able to devote to acquiring that information. In order for an agent to act sensibly in such a domain, it must be able to balance its need for information with the benefit this information provides. It must be able to take advantage of all available information and still act sensibly when less is available. Our multi-fidelity behaviors approach provides a framework which allows an agent's performance to respond aggressively to changes in the quality of available information.

The remainder of this paper is organized as follows. We first explain and discuss our multi-fidelity behaviors approach. Then we describe our application of this approach to our RoboCup-99 Sony legged robot team and discuss the strategy we use to balance the costs and benefits of information for the robots.

Multi-Fidelity Behaviors

Dealing with dynamic variations of the cost and availability of resources is a difficult problem that comes up in several areas of computer science. It has been proposed that there is a need in wireless networking for fast algorithms that compute an approximation to the ideal solution. Algorithms that can control the accuracy of their approximations are called *multi-fidelity algorithms* (Satyanarayanan & Narayanan 1999), which inspired our use of the same term for our robotic agent behavior approach.

Such algorithms are also needed to control real-time agent behavior. Our multi-fidelity behaviors approach is a general framework that allows the performance of the system to swiftly upgrade and gracefully downgrade its performance as resource availability varies without disrupting its own activity with potentially frequent behavior changes. We first define general-purpose modes of behavior (Mataric 1992; Brooks 1986). For example, a foraging robot might have three modes of behavior:

1. Search: the robot searches for the desired objects;
2. Acquire: the robot retrieves found objects;
3. Store: the robot returns the foraged objects to some storage location.

We implement modes of behavior at several "fidelity" levels, which correspond to different levels of resource availability. For example, one possible implementation of the store mode of the foraging robot we mentioned above might require that its GPS receiver is working properly. Another

implementation would only require that the agent's compass is working. And another might allow the agent to use the stars to navigate. By implementing the same mode, or task, at different fidelity levels, we allow the system to upgrade or downgrade its performance as resource availability changes without changing its behavior drastically. In our example, the robot would never stop trying to store the foraged objects; the efficiency of its method would simply change.

In addition to switching between implementations of individual modes, an agent must also switch between modes. To do this, continuously monitors information about its state, as do Nilsson's teleo-reactive agent programs (Nilsson 1994). In addition to increasing the performance of individual modes, our approach also allows mode-to-mode transitions to depend on the quality of state information. However, care must be taken to avoid a situation that could lead to oscillation between modes as information quality changes with time. The intention of our approach is to allow efficient response to changes in available state information while not allowing the robot to oscillate between states.

RoboCup-99 Legged Robot League

We applied our multi-fidelity behaviors approach to the robotic soccer domain (Kitano *et al.* 1997) in the Sony legged robot league of RoboCup-99.[1] All teams in the RoboCup-99 legged robot league used the same hardware platform: the Sony quadruped legged robots (Fujita, Zrehen, & Kitano 1999), shown in Figure 1.[2] The robots are fully autonomous and have onboard cameras. Our image processing, localization and control algorithms run on the onboard processor. The robots are not remotely controlled in any way, and, as of now, no communication is possible between the robots. The only information available for decision making comes from the robot's onboard camera and from sensors which report on the state of the robot's body.

Figure 1: The Sony quadruped robot dog with a ball.

The soccer game consists of two ten-minute halves, each begun with a kickoff. In the kickoff, the ball begins in the center of the field, and each team may position its robots on its own side of the field. After each goal, play resumes with another kickoff. Each team consists of three robots. Like most of the other 1998 and 1999 teams (Veloso, Pagello, & Kitano 2000), we divided our team into two attackers and one goaltender. In this paper, we will focus on the attackers.

The field is 280 cm in length and 180 cm in width. The goals are centered on either end of the field, and are each 60 cm wide and 30 cm tall. Six unique landmarks are placed around the edges of the field (one at each corner, and one on each side of the halfway line) to help the robots localize themselves on the field.

The robotic soccer domain is a very appropriate one in which to study our approach. Raw data from the robot's camera must be processed to extract the information used to control the robot's behavior. First, the data are sent to the *vision module*, which reports which objects are seen, with what confidence, and at what direction and distance (Bruce, Balch, & Veloso). This information is then sent to the *localization module*, which reports an estimate of the robot's angle, θ, and of its x and y location on the field, along with the standard deviations $\sigma\theta$, σx and σy (Lenser & Veloso 2000).

To estimate the robot's position, the localization system applies Monte Carlo sampling and sensor-based resetting to data about the position of field landmarks relative to the robot. The data provided by the localization system is not accurate when the robot has not recently seen landmarks. Therefore, to get accurate localization information, the robot must look for landmarks.

Because the robot is legged and cannot walk completely smoothly, the camera experiences pitch and roll while it walks. This causes the images it collects to change significantly from one frame to the next. This causes the vision system's identification of objects and the estimate of their distances and angles to degrade. The localization system depends heavily on accurate information about the landmarks, so our approach, like many previous robotic systems, requires that the robot stop moving while looking for landmarks. The process of stopping and scanning for landmarks usually takes the robot between 15 and 20 seconds.

Because of this, it is very costly for the robot to acquire information about its location on the field. Although it is obvious that this information is very useful to a soccer-playing robot, soccer, like other dynamic domains, is time-critical, so every moment spent looking around is lost time. Opponents can also use the robot's inattention to their advantage.

Implementation

To implement our multi-fidelity behaviors approach with the legged robots, we identified the basic modes of behavior and wrote low-fidelity implementations of each of them. We wrote higher-fidelity implementations of some, but not all of the behaviors because some do not benefit from localization information and others are so urgent that we cannot allow them to collect localization information.

We defined four basic modes of behavior for the attackers:

1. Recover: the robot tries to recover a recently lost ball;
2. Search: the robot searches the field for a lost ball;
3. Approach: the robot approaches the ball;

[1] Our extensive videos of this event provide invaluable illustrative support to the work presented in this paper.

[2] The Sony AIBO robots were commercially sold in March of 1999. But the robots used for the RoboCup-99 competition are not the same as those commercially available. The robots we used are equipped with slightly different hardware, and, unlike the commercial product, are programmable.

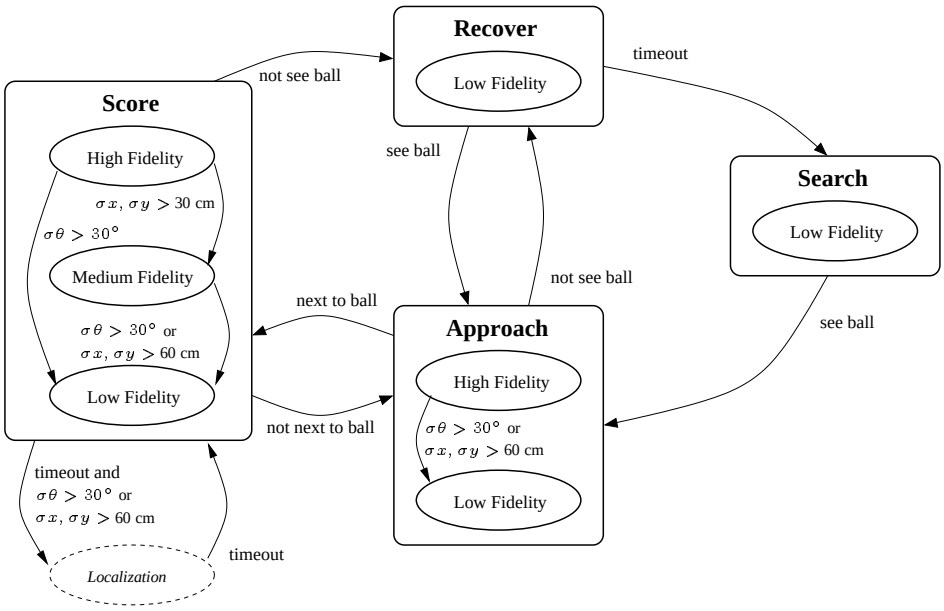

Figure 2: Transitions between the general-purpose modes of behavior and their multi-fidelity implementations. Notice that different behaviors have implementations of different fidelity levels, as appropriate. The transitions are triggered by preset timeouts and processed data sensor, namely visual – see ball, next to ball – and localization information – $\sigma\theta,\ \sigma x,\ \sigma y$.

4. Score: the robot pushes the ball towards the goal.

The robot switches between these modes using information about its state. If the robot does not see the ball and did recently, it tries to recover the ball. If this is unsuccessful, it searches for the ball. If the robot sees the ball, it approaches. If it is close to the ball, it pushes the ball towards the goal. Figure 2 shows an illustration of our algorithm. Two modes (approach and score) are implemented using our multi-fidelity behaviors approach.

Low-Fidelity Behavior

Frequently during games, the standard deviations of the robot's localization information are so high that the information should not be used (see the section, "Balancing the Costs of Information"). As explained previously, it is costly for the robot to stop and look every time its localization information is inaccurate. Therefore, we must make sure the robot can choose actions that will help it achieve its goals even when its localization information is not good enough to use. We will describe the algorithms the robot uses to perform these behaviors with no localization information.

- Recover: The robot often loses sight of the ball while it is trying to manipulate it. Its first strategy is to look around only with its head, but if that is not successful, the robot must move around the field to find the ball.

  With no localization information, the robot cannot use a model of the world to try to return to the point at which the ball was last seen. But because, in most cases, the robot has walked past the ball or pushed it to the side, a very efficient way to recover the ball is to walk backwards. If it still does not see the ball, it reverts to the strategy used by last year's team from Carnegie Mellon: it turns in the direction in which the ball was last seen (Veloso & Uther 1999). After this, the robot considers the ball lost and begins a random search for it.

- Search: When the robot does not know where the ball is, it must wander the field to search for it. Without any localization information, the robot cannot do a systematic "oriented" search of the field. Instead, we wrote a random search algorithm that does not rely on localization information at all. Until the robot sees the ball, it alternates between walking forward a random distance and turning a random angle.

- Approach: Although it would be most efficient for the robot to chart an approach to the ball that would allow it to finish "behind" the ball, facing the opponents' goal, this is not possible without localization information. For that reason, when the robot has no localization information, it approaches the ball by running straight towards it.

- Score: Once the robot has possession of the ball, its strategy is simply to push the ball into the goal. This is difficult without localization information because the robot does not know in which direction to push the ball or how far away the goal is. But our behavior allows the robot to score goals without any information from the localization module. The robot walks sideways around the ball until it sees the goal ahead. It then walks forwards into the ball, pushing it towards the goal.

Higher-Fidelity Performance Enhancements

We built higher-fidelity implementations of two of the robot's behavior modes that take advantage of good localization information. Many performance enhancements are

possible with perfect localization, but those we developed are robust and reliable even with noisy information.

- Approach: If good localization information is available, the robot is able to use its approach to the ball to get into position behind it. If $\sigma\theta < 30°$, then the robot is able to "skew" its approach to the ball, so that when it reaches the ball, it is closer to its goal position behind the ball.
- Score: Based on knowledge of the robot's position, this implementation decides which direction to circle around the ball, or whether to circle at all. If the robot has $\sigma\theta < 30°$ and $\sigma x, \sigma y < 60$ cm, then the robot can choose the shortest direction to circle around the ball. The robot can also determine that it is facing the right direction and, *even if it does not see the goal*, choose not to circle the ball anymore, but to push the ball forwards, towards the goal. This enables the robot to score goals consistently even when it cannot see the goal at all.

  With $\sigma x, \sigma y < 30$ cm, the robot is able to realize whether this would mean circling into a wall. If the robot is trying to get to the other side of the ball, it will choose to circle in the opposite direction. Otherwise, it will choose not to circle at all, but rather to push the ball forward down the edge of the field.

These enhancements allow the robot to score more consistently than it does with the low-fidelity algorithm. The main reason is that, by making the robot's action more efficient, they reduce the amount of time the robot spends moving around the ball. Because the robot's motion is inaccurate and unpredictable, it often taps the ball away while trying to maneuver around it, forcing the robot to stop and search for the ball. Any reduction in the amount of time the robot spends moving near the ball reduces the chances that the ball will be nudged away.

Unenhanced Behaviors

Although we used localization information to improve the performance of two of the robot's modes, we did not improve the other two. There are two reasons for this. Some behaviors do not benefit from localization information. Other behaviors are so urgent that even if they would benefit from localization information, stopping to acquire it would be too expensive.

Behaviors with No Need for Localization We do not allow the robot to acquire or use localization information at all when it is searching for the ball, whether it has lost the ball recently or is conducting a search of the field for it.

Information about the robot's position on the field does not help it in either of these cases. When the ball has recently been lost, the only important information is that it is probably near the robot. Our unenhanced search already takes advantage of this fact. Even when the ball is completely lost, it is no more likely to be in one area of the field than another, so localization information does not help the robot to determine where to look first.

One way of searching for a lost ball is to build an "oriented" search, in which the robot uses localization information to systematically search each area of the field. This relies on very accurate localization information which takes a lot of time. For comparison, we built an oriented search which uses localization information to walk a circuit of the field. Even on an empty field, in which obstacles do not block the robot's view of landmarks, the random search allows the robot to canvass the field more quickly than the oriented circuit search because the robot never has to stop to look for landmarks.

Urgent Action There are two situations in which we allow the robot to use what localization information it does have, but do not allow it to stop to get more. In these cases, swift action is essential, so there is no time for the robot to stop and look for landmarks.

- Approach: Although we have enhanced the approach mode with a skew feature to allow the robot to position itself behind the ball more efficiently, we do not allow the robot to stop during its approach to scan for landmarks.

  This strategy has negatives, clearly. If the robot does not know where it is on the field, it will not know what to do with the ball when it gets to it. Nevertheless, it is better for a robot to look around when it is in possession of the ball than when it is farther from the ball. When the robot is standing near the ball, it is blocking one side of the ball from visibility and attacks, and is able to respond more quickly to an attack because it is already close by.
- Kickoff: We do not allow the robots to localize during the initial kickoff of the game, because a large advantage is gained by succeeding to push the ball into the opponents' side of the field. When the ball moves to one side of the field, it is very difficult for the robots to move it to the other side of the field.

  Instead of relying on localization information, we take advantage of the information we already have: the robots begin the kickoff behind the ball, facing the opposite goal. When the game begins, the robots charge forward into the ball and try to run with the ball for almost half the length of the field (or almost all the way to the opponents' goal). Stopping to localize would give the opponent a good chance to win the kickoff.

Balancing the Costs of Information

We have already described how we adapt to varying levels of localization information. We will now discuss how we balance the cost and benefits of good localization.

In every domain, system designers must strike a different balance between the costs of acquiring resources and the benefits of using them. Even in the Sony robotic soccer domain, different teams came to different conclusions about how much time should be spent acquiring localization information. This year's team from LRP University in France (Bouchefra *et al.* 1999), for example, chose to localize the robot very infrequently, if at all. However, the benefits of accurate localization are significant.

We use a two-constraint system to balance the cost and benefits of good localization information. One constraint ensures that the robot spend a sufficient amount of time acting. The other ensures that if the robot's localization information

quality falls below a preset threshold, the robot will not use it and will stop to look for landmarks as soon as the time constraint allows it to.

We ran two experiments to determine the best values for these two constraints. We tested how long it would take one robot on an open field to score a goal. If the robot took longer than fifteen minutes, we stopped the trial and recorded its time as fifteen minutes.

We positioned the robot at a fixed point, (700, 450) in our coordinate system. This corresponds to 3/4 of the way down the field from the yellow goal, and 1/4 of the way from the right side wall (if facing the yellow goal). We oriented the robot at -135° in our coordinate system, or 45° to the left of facing straight towards the yellow goal. We placed the ball directly in front of it, at the midpoint of the width of the field. We timed how many minutes it took the robot to push the ball into the yellow goal.

Figures 3 and 4 show the results of the two experiments we ran. Each trial is represented by a small tick. The mean of each set of trials is indicated as a bold tick. The grey bar around the mean is one standard deviation.

The experiments have, as we expected, a very high variance, which reflects the challenge of the control task inherent in the robots. The results from the robot's sensors have non-neglible variance and the motion of the robot is unreliable. Also, the statistical localization inevitably gives different results even with exactly the same inputs. We believe that most of the variance is due to the high rate of error in the robot's motion. This unreliability is magnified when the ball is involved. Interestingly, when the robot tries to push the ball forward, it often ends up pushing it at an angle or out to the side, or even walking past the ball. When it walks around the ball, it often accidentally taps it, sending the ball off in an unpredictable direction, as described previously.

Constraint 1—Enforcing Action

Our first constraint requires the robot to spend a certain amount of time acting before it stops to look for landmarks. This is crucial for two reasons. The first we mentioned before: soccer is time-critical, so the robot should only localize as much as is necessary. But we must worry about more than the percentage of time the robot spends localizing versus acting. We must also ensure that the robot does not interrupt its action too frequently. Each time the robot stops to look for landmarks, there is some chance that it will have trouble finding the ball when it finishes localizing and looks for it again. This happens because the robot accidentally nudges the ball away, because it fails to stop moving before looking away from the ball or because another robot steals the ball from it. Stopping more frequently increases the chance that this will happen and the robot will have to begin searching for the ball, a time-consuming procedure.

Our scheme uses a counter to require the robot to act for a specified amount of time before looking for landmarks. The amount of time the robot must act before looking could depend on the confidence the robot has in its current localization information or on its current goals. In our scheme, however, it is invariant.

We require that the robot act for the time it takes the image module to proccess 350 frames of data, or about 40 seconds. Recall that stopping to look for landmarks takes the robot between 15 and 20 seconds, not counting the time it takes it to recover the ball afterwards. So we demand that it spend about $2/3$ of its time acting.

We chose to count time in image frames processed by the vision module because a full system call is more time consuming than using this information. The number of frames per second is constant (8), and since each processed frame invokes an update in the control module, the cost of updating counters for each processed frame is negligible.

We conducted several tests to discover how long we should force the robot to act before looking for landmarks. Figure 3 shows the results of ten trials of each of three different time intervals that we considered viable.

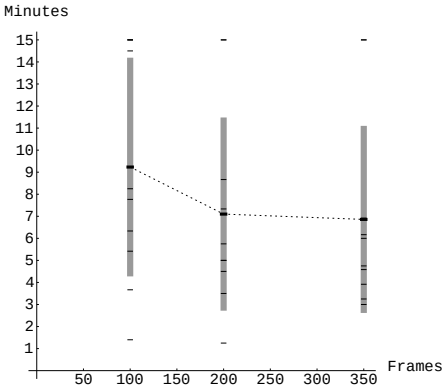

Figure 3: Time taken to score a goal versus how long we require the robot to act before looking for landmarks.

The standard deviations of the trials are, as shown, very high. Nevertheless, there is a clear penalty for localizing too frequently, as shown in the results for the 100-frame (or about 13 second interval). This is because so much stopping to look disrupts the robot's activity. It loses sight of the ball more frequently, and must stop to look for it. Also, scanning for landmarks so often simply takes a lot of time.

Although our results show that 350- and 200-frame intervals (about 40 seconds and 25 seconds, respectively) are roughly equivalent, we chose to use a 350-frame interval. In an actual game, the penalty for stopping more often is much higher than when there is only one robot on the field; when a robot stops to look for landmarks, other robots have a chance to take the ball away.

Constraint 2—Sufficient Localization

The second constraint is how accurate we demand the localization information to be. We measure accuracy with the standard deviations returned by the localization module. If the information is accurate enough, the robot should not stop to look for landmarks when the timing constraint allows it to. But if the localization information is not accurate enough, the robot should not use it.

It is not immediately obvious how good our localization information must be before it is usable. Clearly, if our de-

mands are too high, the robot will rarely be able to use the information it has gathered. And if they are too low, it will use information that is so inaccurate as to be useless at best and damaging at worst.

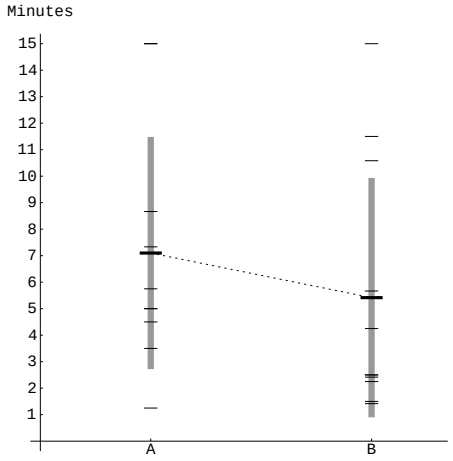

Figure 4: Time taken to score a goal in two setups for transitioning among the multi-fidelity score behavior. Setups A and B correspond to different standard deviation bounds of the localization values for the robot's orientation, θ, and its x, y location on the field.

Figure 4 shows the results of ten trials each of two different sets of standard deviation bounds, labelled A and B. In setup A we set $\sigma\theta < 15°$ and $\sigma x, \sigma y < 30$ cm; in setup B, we set $\sigma\theta < 30°$ and $\sigma x, \sigma y < 60$ cm.

If we demand the localization values be too accurate before allowing the robot to use them, it is often unable to use them and must revert to our low-fidelity strategy, detailed previously. If it is still able to use them sometimes, this just slows it down, as seen in Figure 4, setup A. Other experiments we have run have shown us that if the robot must rely too heavily on the low-fidelity strategy, it is much more likely to tip the ball in the wrong direction and accidentally make an own goal. But, if we demand too little accuracy from the localization values, the robot will rely on them even when they are faulty. We ran other experiments which showed that this too causes the robot to score own goals.

We chose to use the standard deviation values used in setup B from Figure 4 ($\sigma\theta < 30°$, $\sigma x, \sigma y < 60$ cm), because, in our experiments, there was a clear advantage to these settings. These standard deviation bounds mean that the 95% confidence interval for the x and y location of the robot is ± 120 cm, or almost the entire field. These experiments show that, although it is crucial to have a rough estimate of angle, it is not important for the robot to know where it is on the field.

Conclusion

In this paper, we have described our multi-fidelity behaviors approach to designing behaviors for resource-poor real-time environments. We briefly described the approach in general terms. We then elaborated on our implementation of this approach in the RoboCup-99 Sony legged robot league, describing our breakdown of the robot's behavior into modes corresponding to multi-fidelity behaviors. Finally, we presented our two-constraint technique for balancing the cost and benefits of localization information with the Sony robots, along with the results of experiments we ran to determine the best values for those constraints. We are applying the multi-fidelity behaviors approach to several of our other robotic systems.

Acknowledgements

We would like to thank Sony for providing us with wonderful robots and walking movements to work with. We would like to thank Jim Bruce for developing the vision system and Scott Lenser for developing the localization algorithm for our robots. Thanks to Tucker Balch for helpful discussions and suggestions on this work.

This research is sponsored in part by the Defense Advanced Research Projects Agency (DARPA) and the Air Force Research Laboratory (AFRL) under agreement numbers F30602-97-2-0250 and F30602-98-2-0135. The views and conclusions contained in this document are those of the authors and should not be interpreted as necessarily representing official policies or endorsements, either expressed or implied, of the Air Force or the United States Government.

References

Bouchefra, K.; Hugel, V.; Blazevic, P.; Duhaut, D.; and Seghrouchni, A. 1999. Situated agents with reflexive behavior. In *Proceedings of IJCAI-99 Workshop on RoboCup*, 46–51.

Brooks, R. A. 1986. A robust layered control system for a mobile robot. *IEEE Journal of Robotics and Automation* RA-2:14–23.

Bruce, J.; Balch, T.; and Veloso, M. Fast and inexpensive color image segmentation for interactive robots.

Fujita, M.; Zrehen, S.; and Kitano, H. 1999. A quadruped robot for RoboCup legged robot challenge in Paris'98. In Asada, M., and Kitano, H., eds., *RoboCup-98: Robot Soccer World Cup II*. Berlin: Springer Verlag. 125–140.

Kitano, H.; Asada, M.; Kuniyoshi, Y.; Noda, I.; and Osawa, E. 1997. RoboCup: The robot world cup initiative. In *Proceedings of the First International Conference on Autonomous Agents*.

Lenser, S., and Veloso, M. 2000. Sensor resetting localization for poorly modelled mobile robots. In *Proceedings of ICRA-2000, the International Conference on Robotics and Automation*.

Mataric, M. J. 1992. Behavior-based control: Main properties and implications. In *Proceedings of the IEEE International Conference on Robotics and Automation, Workshop on Architectures for Intelligent Control Systems*.

Nilsson, N. J. 1994. Teleo-reactive programs for agent control. *Journal of Artificial Intelligence Research* 1:139–158.

Satyanarayanan, M., and Narayanan, D. 1999. Multi-fidelity algorithms for interactive mobile applications. In *Proceedings of the 3rd International Workshop on Discrete Algorithms and Methods for Mobile Computing and Communications*.

Veloso, M., and Uther, W. 1999. The CMTrio-98 Sony legged robot team. In Asada, M., and Kitano, H., eds., *RoboCup-98: Robot Soccer World Cup II*. Berlin: Springer Verlag. 491–497.

Veloso, M.; Uther, W.; Fujita, M.; Asada, M.; and Kitano, H. 1998. Playing soccer with legged robots. In *Proceedings of IROS-98, Intelligent Robots and Systems Conference*.

Veloso, M.; Pagello, E.; and Kitano, H., eds. 2000. *RoboCup-99: Robot Soccer World Cup III*. Berlin: Springer Verlag. To appear.