RoboCup as an Introduction to CS Research

Peter Stone

Department of Computer Sciences
The University of Texas at Austin
1 University Station C0500
Austin, Texas 78712-1188
pstone@cs.utexas.edu
http://www.cs.utexas.edu/~pstone

Abstract. This paper proposes using topics central to RoboCup, particularly autonomous agents and multiagent systems, as the subject-matter for a course designed to introduce undergraduate students to all facets of computer science research. Experiences are presented from the design and implementation of such a course. The course is structured around an ongoing incremental programming project that culminates in a class tournament in the RoboCup Soccer Server, an open-source infrastructure built to support multiagent systems research and education.

1 Introduction

Most upper-division computer science (CS) majors have determined that they enjoy taking CS classes, or at least that they are relatively good at it. However, this determination may not be indicative of a propensity for computer science *research*. Indeed, many CS Ph.D. candidates have discovered that they are *not* cut out for research only after investing several years in a graduate program. Conversely, there are presumably those students who could have enjoyed quite successful research careers had they only thought to give it a shot.

One obvious explanation for these phenomena is that traditional undergraduate computer science courses demand very different skills from those required of researchers[1]. In particular, traditional coursework typically requires students to:

- read textbooks;
- sit (often silently) in large lectures;
- execute programming tasks with correct and complete answers;
- work alone; and
- take exams.

On the other hand, researchers typically must:

- read about and critically assess original research;

[1] A similar argument likely holds in other areas of scientific research. This paper speaks specifically from a computer science perspective due to the fact that the author's experience lies mainly in that field.

– speak in public;
– collaborate effectively with peers;
– devise solutions and/or approaches to open-ended problems; and
– write about these solutions and/or approaches.

Given these differences, it is not surprising that proficiency at coursework does not correlate perfectly with proficiency at research. In response to this observation, many Ph.D. programs now encourage and/or require their students to engage in research activities from the outset of their graduate studies with the aim of helping students to determine quickly whether they are truly interested in pursuing a career in research.

This paper advocates giving students an opportunity to make this determination before even entering graduate school. One method for doing so is to encourage students to engage in individual undergraduate research projects. However such opportunities are quite varied in scope and limited in availability. Another method for doing so—the one put forth in this paper—is to offer undergraduate coursework specifically designed to require the same skills required of professional researchers.

This paper argues that RoboCup is an ideal topic around which to build such a course. Experiences are presented from the design and implementation of such a course focusing in particular on *autonomous agents and multiagent systems*. The course is structured around an ongoing incremental programming project culminating in a class tournament in the RoboCup Soccer Server, an open-source infrastructure built to support multiagent systems research and education [11].

The remainder of this paper is organized as follows. Section 2 presents some general principles to be followed by a course designed to introduce research to undergraduates. Section 3 details a specific implementation of these principles in the form of a course entitled *Autonomous Multiagent Systems*. Section 4 presents anecdotal results from this course and concludes.

2 General Principles

This section proposes a set of five principles to be followed when creating a course for the purpose of introducing undergraduates to scientific research. These principles are illustrated in the context of an AI-based course that makes extensive use of the RoboCup domain in Section 3.

Open-ended project. The most crucial element of such a course is a long-term open-ended assignment that leaves as much room as possible for creativity and innovation. For practical purposes and to avoid mass frustration on the parts of the students, it must be possible to get *some* result fairly easily. However it is very important that there be no natural stopping point. Instead, students should propose an approach and a goal for their projects on their own before beginning.

Students are used to assignments with the characteristic that they can recognize when they have reached a correct or sufficient result and thus can safely quit. However, researchers often work on long-term projects with no clear answers. Thus it is important to have some such challenging project in the class. The difficulty then becomes motivating the students to meet the challenge.

One motivational technique is to have the project culminate in a class competition. Even if the competition has no impact on the student's grade, it can be a strong impetus to going above and beyond the effort typically reserved for a class project. AI is rich in domains that are appropriate for such competitions, with the RoboCup soccer domain [11] being perhaps the most popular example. Experiences with this domain are reported in Section 3.

Read research papers. Students are used to reading secondary sources, such as textbooks, in which many ideas are synthesized and put into a coherent perspective. While these sources may be preferable to primary sources from a pedagogical perspective, they do not prepare students for the activity of reading and assessing cutting-edge research papers.

Consequently, course readings should be predominantly primary sources. They need not be the most current articles in the field. But they should be assigned in the form in which they were originally published. Whenever possible, readings with opposing views should be selected so that the students must form their own opinions about their relative merits.

Finally, the articles must be chosen so as to be accessible to students without extensive background (other than previous course readings). This requirement is perhaps the most difficult to meet for many course topics.

Require class participation. It is not sufficient to merely *assign* the research papers. There must be some incentive for the students to read them from a critical perspective. For example, students can be required to come to class prepared with questions about the readings, or brief written reactions to the readings can be required.

Whatever the method, the goal should be to have every student verbally express reactions to the readings. Especially in its more philosophical guise, AI is rife with topics that excite stimulating class discussions.

Foster improved scientific writing. By all accounts, writing is perhaps the skill most lacking in students today, perhaps particularly so for science majors. As such, it is essential that the course place a heavy emphasis on scientific writing. In particular, the students should at least be required to write a report of their project in the form of a research paper. This report should count for a significant part of their final grades.

In addition to the final report, there should be at least one prior writing assignment so that students can receive, and act on feedback from the instructors. Ideally, this feedback should concern style and organization as well as content.

Encourage collaboration. Students are often made to complete their work entirely on their own. They may ask their instructors for hints or help, but they are usually not allowed to work with their classmates. In contrast, a glance at the bylines in the literature shows that most research papers are the result of collaborations. Thus, especially for the open-ended project, students should be given the opportunity to work in small groups.

Surely there are many ways in which the above principles can be implemented in the class. Indeed, the details are necessarily dependent upon the subject being taught. Section 3 illustrates these principles in concrete form within a computer science curriculum.

3 Implementation

The main thesis of this paper is that RoboCup is an ideal topic around which to build a course that implements the principles laid out in Section 2. The main advantages of RoboCup are that it it is an appealing domain for students of both genders; it is a popular testbed domain used by active researchers in the field with many recent papers published pertaining to it; it admits for a wide variety of research foci; and it lends itself well to exciting, visual competitions. No other domain known to the author combines all of these aspects as well as does RoboCup.

This section details a specific implementation of the general principles in the form of a course focusing on autonomous agents and multiagent systems (AAMAS). AAMAS is one of the fields to which RoboCup participants have contributed consistently and prominently over the years. Despite being the basis for a large subfield of AI, there is no generally accepted definition of artificial intelligence *agents*. In loose terms, agents are programs that (i) sense their environment, (ii) make decisions about how to act based on these sensations, and (iii) then execute these actions. Autonomous agents do all three of these steps on their own, i.e. without a human in the loop. Multiagent systems are collections of multiple agents that interact with one another. The field of AAMAS covers a wide variety of research foci and applications, including software-based information processing, robotic control of multiple agents, entertainment agents, and tutoring agents [6].

The course described in this section is called *Autonomous Multiagent Systems*. It has been taught twice, most recently during the fall of 2002[2]. It will be offered again during the spring of 2004 [3]. The course provides a broad introduction to autonomous agents with an emphasis on multiagent systems. Topics include:

- agent architectures;
- inter-agent communication;
- teamwork;
- distributed rational decision making;
- agent modeling;
- multiagent learning; and
- entertainment agents.

[2] `http://www.cs.utexas.edu/~pstone/Courses/378fall02`
[3] `http://www.cs.utexas.edu/~pstone/Courses/378spring04`

This paper will not go into detail about the subject matter of the course except as far as is necessary for the purpose of clarity. The subject matter is in a sense subordinate to the main purpose of the course. Students are told explicitly that in order to succeed they will need to attain a mastery of the subject. However evaluation is based primarily on their ability to engage in the full range of activities required of researchers.

The remainder of this section describes how each of the principles presented in Section 2 is implemented in the course.

3.1 Project

The central focus of the course is a semester-long build-up towards a class robotic soccer competition in the RoboCup Soccer Server [10]. The Robot Soccer World Cup, or RoboCup, is an international research initiative that uses the game of soccer as a domain for artificial intelligence and robotics research. The RoboCup Soccer Server, coupled with the large body of available client code, is an infrastructure that is designed to be appropriate for both scientific research and education [11].

Soccer Server is a multiagent environment that supports 22 independent agents interacting in a dynamic, real-time environment. The server embodies many real-world complexities, such as noisy, limited sensing; noisy action and object movement; limited agent stamina; and limited inter-agent communication bandwidth. AI researchers have been using the Soccer Server to pursue research in a wide variety of areas, including real-time multiagent planning, real-time communication methods, collaborative sensing, agent/opponent modeling, and multiagent learning [2].

In addition to the server itself being publicly available under an open-source paradigm, users have contributed several clients that can be used as starting points for newcomers to the domain. One example is the CMUnited simulated soccer team [15], champion of the RoboCup-98 and RoboCup-99 robotic soccer competitions. After winning the competitions, much of the CMUnited source code became publicly available, and several groups used it as a resource to help them create new clients for research and as entries in subsequent competitions. Figure 1 shows a screen shot of Soccer Server.

Soccer Server and the CMUnited client code are *widely* and *freely* available over the internet using an *open source* paradigm. The software is *packaged* for easy installation, *supported* both by the developers and by the large *community* of current users.

This infrastructure is a *comprehensive, implemented* MAS designed for *simulation* experiments. It consists of several independent *components*, including visualization, sample client, and coach modules. The coach module is often used as a tool for *experiment construction.* The most natural and compelling form of *measurement* is game results in tournaments with multiple teams, but the infrastructure also includes *data collection and analysis* tools for more rigorous scientific measurement.

Judging by the large user community (over 1000 researchers worldwide), this infrastructure is very *usable*; the fact that it has been successfully used

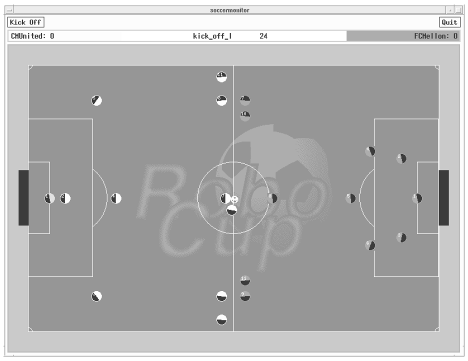

Fig. 1. Window image of Soccer Server

for multiple international competitions is a testament to its *robustness*. New users can take advantage of its *progressive complexity* by starting with a single agent and gradually increasing the size of teams and their communicative and organizational capabilities. A recent addition to the infrastructure is the ability to induce *intentional failures* by disabling selected players.

The italicized words above are all characteristics identified by Gasser [5] as essential or desirable for MAS infrastructures that support science and education.

Two additional favorable properties of Soccer Server are that it is both an appealing domain for students of both genders [14] and a popular testbed domain used by active researchers in the field. An IJCAI-97 challenge paper [8] identified three general research challenges that can be addressed within Soccer Server as being

- multiagent learning;
- teamwork structures; and
- agent/opponent modeling.

As laid out in [15], other relevant research issues include inter-agent communication in single-channel, low-bandwidth environments; coordination with limited communication, collaboration in a dynamic real-time environment; organizational structures; distributed sensing/sensor fusion; resource management; agent monitoring; and multiagent planning. These research topics are all addressed by various researchers in the continuing series of RoboCup books [1, 2, 7, 16, 17].

In addition to being the substrate domain for much original research, Soccer Server has been used previously as a basis for a few other university courses (e.g. [4, 19]). These courses have previously focussed primarily on teaching AI and/or multiagent systems rather than introducing CS research in general.

Students in the course described here are assigned a series of four preliminary programming assignments designed to get them familiar with Soccer Server and the CMUnited client code. By the end of these preliminary assignments, they have created a fully functional team, but not one that is particularly competent.

The students are then encouraged to propose an improvement on this team as the topic for their final projects[4]. The improvement need not be for the purpose of creating a winning team in the final tournament. For example, one student proposed to use machine learning techniques to learn a good goaltender without paying attention to the rest of the team. In practice, about half of the students make earnest attempts to create top quality teams, while the remainder focus on interesting side issues. Both types of projects can yield (and have yielded) quality results.

3.2 Readings

The majority of the readings for this course are primary sources chosen both to introduce particular topics relevant to the course and to engender some controversy. For example, during one of the early weeks, the students learn about "agent architectures" by reading both an article espousing completely reactive agents (e.g. [3]) and an article that argues in favor of using more deliberative agents (e.g. [13]).

In order to encourage the students to complete the readings in a timely fashion, they are required to submit a brief written answer to a single question pertaining to the readings at least 2 hours before the class starts. For example, a question associated with the above readings has been:

> Part one of your current programming assignment can be implemented reactively. Describe a non-reactive soccer-playing behavior.

Associated with readings on "agent modeling" was the following:

> Describe a domain not mentioned in the readings in which agent modeling could provide a benefit.

Note that these questions can be answered briefly and have no right answers. But in order to respond, students must complete and understand the readings. The fact that the responses are due two hours before class allows the instructor to incorporate them into the class discussion.

3.3 Class Participation

Another effect of the questions pertaining to the readings is that the students *do* come to class prepared to discuss the readings. There have been many extended and heated class discussions pertaining to specific aspects of agents and multiagent systems.

Another important component of class participation is that each student is required to moderate at least one class discussion pertaining to that week's readings. The student discussion is specifically *not* a presentation of the readings:

[4] They are also given the option to propose a programming project in a multiagent domain of their choice, but typically few students choose to do so.

the students are encouraged to assume that the class members have all done the readings. Rather, they are instructed to either defend a controversial statement or pose a question and be prepared to defend either side depending on how the class reacts. Their explicit goal is to moderate an interesting and extended discussion.

This activity turns out to be one of the most difficult for the students to complete. Many of them are not used to speaking in front of a class, and they have rarely been put in the position of *facilitating* discussions as opposed to defending specific positions. Some examples of successful questions generated by the students are:

- "Is it better to build in teamwork at the architecture level, or to first build individually functioning agents and then add on teamwork constructs?"
- "We've seen some examples of machine learning agents that spend lots of time and effort learning, but don't do as well as hand-coded agents. Is it worth all the trouble?"
- "Do we want machines (programs, appliances) to have emotions and personality?"
- "Nash Equilibrium is limited in its applicability due to its exponential complexity and the inability to give a deterministic solution. Agree or Disagree?"

3.4 Writing

The course requires a good deal of writing from the students. As already mentioned, the students are required to provide weekly written responses to questions related to the readings. They receive feedback pertaining to the clarity and soundness of their responses.

Much more significantly, the students are required to write three written documents pertaining to their final projects. First, they write project proposals defining their goals for their projects as well as the proposed methods for achieving them. Second, they revise their proposals and add a section on their work in progress to create progress reports. Finally, they write final reports in the format of conference papers.

In practice, these writing assignments often show an evolution of the students' ideas. For example, a common proposal is to use some machine learning technique that has been mentioned in the readings to improve some aspect of the soccer team's performance. However, based on feedback from the instructors and their initial attempts, the students often realize that their initial proposal was too ambitious and scale it back to more realistic levels given the time available. As one student who was attempting to use machine learning in conjunction with a simple general agent architecture wrote in the final report: the "machine learning part has proven difficult and elusive, and has turned into a research project over suitable data, choice of representation, and machine learning algorithms." Nonetheless, there have been many modestly successful uses of machine learning in the student teams.

3.5 Collaboration

Productive collaboration cannot be forced. However it should be encouraged. In this course, students are given the opportunity, but are not required, to work in pairs on the final project. Teams of two must still write their proposals and reports individually, with clear indications of what role each person played in the collaboration; and as such, more is expected from them as a final product.

The robotic soccer project lends itself to such collaboration nicely since there are many different ways in which the students can divide up the work. For example, one can focus on offense while the other focuses on defense; one can focus on low-level skills while the other focuses on team strategy; one can work on agent development tools while the other focuses on using them to create the team; etc. Each of the these three task division strategies has been used by students in the class.

3.6 The Tournament

The class culminates in the class tournament. The students are told at the outset that performance in the tournament will have no negative impact on their grades (A strong performance *can* have a positive impact.). Nonetheless, the tournament is a strong motivational factor for the students. Visitors are invited to the event and the students present their approaches orally and field questions as their teams are playing.

The performance spread among the teams is often very large, especially given the fact that some students do not focus on creating winning teams. In the end, all class champions have been tested against a mid-range RoboCup entry and lost significantly: despite starting with a fairly detailed client code base, the students are not able to attain competitive world-class levels. However, given their time limitations this fact is neither surprising nor discouraging.

4 Results and Conclusion

The true measure of this course is the degree to which it has influenced students' decisions to pursue a career in computer science research, either positively or negatively. Perhaps it would be possible to devise a controlled study to measure a relevant statistic of this form. However, such a study has not been done and is beyond the scope of this paper.

There is, however, anecdotal evidence available, at least from the positive perspective. Both times the course has been run, at least one student has described the course in graduate school applications as a primary motivation for going on to do research. In addition, two students from the Fall 2002 version of the course actively contributed to the UT Austin entry in the RoboCup 2003 competition. In one case, the student played a key role in the winning entry in the on-line coach competition. The research involved may lead to this student's senior thesis.

Various course evaluations and surveys have also provided evidence that the students appreciate the opportunity to be exposed to the various components of scientific research. Some of the comments from students have included:

– “I really like reading from research literature. Just the fact that it’s actual research provokes curiosity;”
– “The discussions we have in class are quite unique, I haven’t had such involving discussions in any class before;”
– “Format of the class is perfect. I’ve waited through three years of college for a class like this. . . . I like the class so much that my other classes now disappoint me;” and
– “I loved this course, and it’s the best possible way to be exposed to AI.”

Another indication that the course has been successful is that many of the students’ final projects have addressed interesting research topics. Of the 25 students who have successfully completed the course to date, 24 have done their final projects in the Soccer Server domain[5].

A rough breakdown by topic follows. The list does not account for exactly 24 projects because some projects (4) were done by pairs of students, and some are counted under more than one topic.

– Five projects made use of *machine learning* in some way. Two of these used reinforcement learning to learn aspects of the team, in one case just the goalie, and in another case, the entire defensive strategy. A third project learned models of player and ball motion so as to predict their positions when they are not visible to the player. One project aimed towards learning to recognize and predict breakaway situations, and another used on-line adaptive formations via opponent modeling. This last was the winner of the class tournament in 2002. During the competition, it displayed an ability to adjust its players’ positions on the field appropriately during play.
– Four projects examined the efficacy of some general *agent architectures* in the robotic soccer domain. Specifically, they used a BDI model, cooperative behavior nets, a subsumption architecture, and a decision tree represenation.
– Three projects incorporated aspects of *ant-based* system design as presented by Parunak [12]. One modeled the entropy flow through the system. One built an ant-like system using the subsumption architecture and then attempted to improve the result via machine learning. One focused on finding the minimal reactive rules necessary to create a functioning team.
– Three projects focussed on inter-player *communication*. One of these developed a paradigm for communincation of high-level strategies. Another enabled players to communicate their world states. The other focussed on the challenge of quickly propagating information from a decision-maker to the rest of the team despite the limitations on message bandwidth enforced by the simulator.
– Three projects focussed on improving high-level general soccer *strategies*, including players’ roles, teams’ formations, etc.
– Three projects were geared primarily towards creating *winning teams*. In these cases, a wide range of challenges were addressed, often with a good

[5] The other did an interesting study of a multiagent-based approach to neural network formation using the Swarm package [9]

deal of heuristics and manual tuning of parameters. In general, low-level behaviors were the main emphasis. The teams resulting from these projects did indeed do well in the class tournaments, winning in 2001 and finishing 2nd and 3rd in 2002.

- Two projects implemented *planning* approaches to the Soccer Server domain. In both cases, planning was done dynamically during the course of the game. In one case, each individual constructed entire team plans and then only executed its own part of the best team plan.
- One ambitious project focussed on implementing the team from scratch (i.e. without the benefit of any code base). Whereas most projects have been programmed in C++ or Java due to the existing code bases in those languages, this project was programmed in Perl.
- One project facilitated short-term real-time coordination among a few agents for the purpose of executing certain *combination plays*, such as the "give-and-go" and the centering cross.
- One project addressed the challenge of maintaining an accurate world state despite only seeing a part of the environment at any given time. The approach was to predict the movements of other agents and the ball, and in the absence of visual information to update the player's world state accordingly.

Several of these projects were well-written and interesting enough to warrant follow-up experiments and (usually with a fair amount of additional effort) could possibly have led to eventual publications. However, so far, none of the reports has been extended in such a way.

All of the course materials are available on-line, including the RoboCup soccer server system and the CMUnited client code. As such, the course should be fully repeatable at other institutions. One goal of this paper is to fully describe the motivations behind the course so as to encourage such repetition. The primary goal has been to illustrate the successful use of a course on autonomous agents and multiagent systems, with project assignments in the RoboCup Soccer Server, as an introduction to CS research.

In the future, the course may be extended to include exposure to real robots as well as the Soccer Server. During the spring of 2003, the author taught a graduate level course [6] that culminated in a class entry in the RoboCup Sony four-legged robot league [18]. The opportunity to work with these legged robots may be extended to undergraduates in future iterations of the course described in this paper.

Acknowledgements

Thanks to Tucker Balch, Jeremy Cooperstock, Gal Kaminka, Manuela Veloso, and José Vidal for ideas from their related courses. The author also wishes to thank Vinay Sampath Kumar, a TA for the 2002 version of the course, as well as the NYU and UT Austin students who have participated in the course.

[6] `http://www.cs.utexas.edu/~pstone/Courses/395Tspring03`

References

1. Minoru Asada and Hiroaki Kitano, editors. *RoboCup-98: Robot Soccer World Cup II*. Lecture Notes in Artificial Intelligence 1604. Springer Verlag, Berlin, 1999.
2. Andreas Birk, Silvia Coradeschi, and Satoshi Tadokoro, editors. *RoboCup-2001: Robot Soccer World Cup V*. Springer Verlag, Berlin, 2002.
3. Rodney A. Brooks. Intelligence without representation. *Artificial Intelligence*, 47:139–59, 1991.
4. Silvia Coradeschi and Jacek Malec. How to make a challenging AI course enjoyable using the RoboCup soccer simulation system. In Minoru Asada and Hiroaki Kitano, editors, *RoboCup-98: Robot Soccer World Cup II*. Springer Verlag, Berlin, 1999.
5. Les Gasser. Mas infrastructure definitions, needs, and prospects. In *Proceedings of the Autonomous Agnets 2000 Workshop on Infrastructure for Scalable Multi-Agent Systems*, Barcelona, Spain, June 2000.
6. Maria Gini, Toru Ishida, Cristiano Castelfranchi, and W. Lewis Johnson, editors. *Proceedings of the first international joint conference on Autonomous agents and multiagent systems*. ACM Press, 2002.
7. Hiroaki Kitano, editor. *RoboCup-97: Robot Soccer World Cup I*. Springer Verlag, Berlin, 1998.
8. Hiroaki Kitano, Milind Tambe, Peter Stone, Manuela Veloso, Silvia Coradeschi, Eiichi Osawa, Hitoshi Matsubara, Itsuki Noda, and Minoru Asada. The RoboCup synthetic agent challenge 97. In *Proceedings of the Fifteenth International Joint Conference on Artificial Intelligence*, pages 24–29, San Francisco, CA, 1997. Morgan Kaufmann.
9. N. Minar, R. Burkhart, C. Langton, and M. Askenazi. The swarm simulation system: A toolkit for building multi-agent simulations, 1996. available at `http://www.santafe.edu/projects/swarm/overview/overview.html`.
10. Itsuki Noda, Hitoshi Matsubara, Kazuo Hiraki, and Ian Frank. Soccer server: A tool for research on multiagent systems. *Applied Artificial Intelligence*, 12:233–250, 1998.
11. Itsuki Noda and Peter Stone. The RoboCup soccer server and CMUnited clients: Implemented infrastructure for MAS research. *Autonomous Agents and Multi-Agent Systems*, 7(1&2), July 2003. To appear.
12. H. Van Dyke Parunak. "go to the ant": Engineering principles from natural agent systems. *Annals of Operations Research*, 75:69–101, 1997.
13. Reid Simmons. Structured control for autonomous robots. *IEEE Transactions on Robotics and Automation*, 10(1):34–43, February 1994.
14. Elizabeth Sklar, Amy Eguchi, and Jeffrey Johnson. Robocupjunior: learning with educational robotics. In Gal A. Kaminka, Pedro U. Lima, and Raul Rojas, editors, *RoboCup-2002: Robot Soccer World Cup VI*. Springer Verlag, Berlin, 2003.
15. Peter Stone. *Layered Learning in Multiagent Systems: A Winning Approach to Robotic Soccer*. MIT Press, 2000.
16. Peter Stone, Tucker Balch, and Gerhard Kraetszchmar, editors. *RoboCup-2000: Robot Soccer World Cup IV*. Lecture Notes in Artificial Intelligence 2019. Springer Verlag, Berlin, 2001.
17. Manuela Veloso, Enrico Pagello, and Hiroaki Kitano, editors. *RoboCup-99: Robot Soccer World Cup III*. Springer Verlag, Berlin, 2000.
18. Manuela Veloso, William Uther, Masahiro Fujita, Minoru Asada, and Hiroaki Kitano. Playing soccer with legged robots. In *Proceedings of IROS-98, Intelligent Robots and Systems Conference*, Victoria, Canada, October 1998.
19. José M. Vidal and Paul Buhler. Teaching multiagent systems using RoboCup and biter. *The IMEJ of Computer-Enhanced Learning*, 4(2), 2002.