An Approach to Close the Gap between Simulation and Real Robots

Yuan Xu, Heinrich Mellmann, and Hans-Dieter Burkhard

Institut für Informatik, LFG Künstliche Intelligenz, Humboldt-Universität zu Berlin,
Rudower Chaussee 25, 12489 Berlin, Germany
{xu,mellmann,hdb}@informatik.hu-berlin.de

Abstract. Numerous simulators have been developed over the years to assist robotics research in the development, testing, and evaluation. Nevertheless, there is still a big gap between the simulation and the reality. This makes it difficult to transfer methods and code. The 3D simulator — SimSpark is developed and used by a big community of AI researchers in RoboCup. But up to now there are only few applications to real robots. In this paper, we discuss the general possibilities how the SimSpark simulator can be used to support research in cognitive robotics and present applications on the humanoid robot Nao. As a result of our investigation we have developed a unified team playing both in Simulation League and Standard Platform League in RoboCup.

1 Introduction

The Robot World Cup (RoboCup) initiative is an attempt to foster artificial intelligence and intelligent robotics research by providing a standard problem where a wide range of technologies can be integrated and examined [8]. There are different leagues that make it attractive as an environment for researchers to focus on a set of specific problems on the way to the final goal — "By the year 2050, develop a team of fully autonomous humanoid robots that can win against the human world soccer champion team."

At present, there are several different leagues, which are focus on different subtasks. However, there is a lot of work being repeated in the different leagues while solutions for the same and similar issues exist in another league. For example, self localization, biped locomotion and etc. It is useful to achieve synergy effects for the same challenges in different leagues.

Numerous simulators have been developed over the years to assist in the development, testing, and evaluation of robots. Every robotics research group uses simulators to develop their robots. Designing and implementing a good robot simulator is a difficult and time consuming task, so it makes sense to reuse the existing work. And also, the simulation teams want to apply their solutions to real robot. It makes sense to integrate simulation team and real robot team.

There are already some collaborations between researchers from different leagues. Dylla et al.[5] investigated to model soccer knowledge in a way that they are usable for multiple RobCup soccer leagues. Mayer et al.[11] proposed

N. Ando et al. (Eds.): SIMPAR 2010, LNAI 6472, pp. 533–544, 2010.

a road map to collaboration between Soccer Simulation League and Humanoid League. At the same time, the USARsim[15] simulator is used as bridging tool between the Rescue Real Robot League and the Rescue Simulation League.

In this paper, our investigations concerning simulation and real robot focus on Standard Platform League and Simulation League. Since 2008, the Standard Platform League and 3D Simulation League use the same robot — Nao from Aldebaran Robotics. Our team — Nao Team Humboldt started at 2008 in Standard Platform League, and also participate in 3D Simulation League now. We use a common Core of our program for both platforms. Actually our framework runs on Webots and log simulator, too. But throughout the paper we will concentrate on the real robots and SimSpark — the simulator of 3D Simulation League.

This paper is organized as follows. Section 2 describes the current state of the RoboCup Simulation League and Standard Platform League. The implementation of our team is described in Section 3. Section 4 gives some experimental result. Section 5 discuses the problems which we are working on, followed by conclusion and future work in section 6.

2 Current State of the Leagues

Simulation League and Standard Platform League both have long histories and big communities. In this section, we describe the current state of the two leagues.

2.1 3D Simulation League

Simulation league exists from the very beginning of RoboCup in 1997. The 2D simulator only runs on a relatively abstract environment. The participants concentrate mainly on coordination and cooperation of robot teams. Because of the simplified model of 2D simulation league, a three-dimensional physical simulation[9] was introduced in RoboCup 2004, but only sphere shaped agents ran around the field. Thereby increasing the realism of the simulated environment and making it more comparable to the other RoboCup league environments; but still being able to simulate more players on the field than other leagues. Thus giving rise to higher level research on larger groups of slightly more realistic soccer playing robots.

Since 2007, the humanoid robots are introduced into the 3D Simulation League. This opens up opportunities for research on lower level control of humanoid robots as well as higher level behaviors in humanoid soccer and getting closer to how humans play the game. The participants have to create soccer playing agents for a team which have to deal with the higher complexity of a human shaped body in the (simulated) physical environment. Hopefully, the results can be transferred to humanoid robotics more easily, and our research investigates these connections.

As a consequence of the changes to the 3D simulator, a (temporary) shift of problem solving to the more basic problems of body control could be observed.

This is related to restrictions concerning tactics: how to plan a team play when the basic skills do not allow, e.g., for a good pass.

At the same time, simulation allows for more players than with real robots. In 2010, the field size of the 3D Simulation League increased to 15×10 meters, and the games are played in 6 against 6, see Fig. 1. This means, coordination and cooperation between robots becomes important again like in 2D simulation. It is a very interesting two tired problem: to develop the skills for better tactics on one hand, and to realize tactics based on the actually available skills on the other hand.

Fig. 1. A 6 vs. 6 game is playing in the 3D Simulation League at RoboCup 2010 in Singapore

2.2 Standard Platform League

In the Standard Platform League (SPL) all teams use the same robotic platform. On one hand it gives scientists the opportunity to concentrate on the software development only. On another hand it provides a better possibility to compare the developed algorithm in the game, since no team may gain advantages by changing the hardware of the robot (e.g., using stronger motors). Fig. 2 shows a scene from a SPL soccer game.

Like in the Simulation League the agents (i.e. robots) operate fully autonomous. In particular there is no external control neither by human or by computer. Perception is based on different sensors (cf. Table 1) and calculated by the on board computer. Communication is possible via wireless LAN, but it maybe corrupted by environmental conditions. The on-board computer is also used for planning and control. The runtime system provides special middle ware for the communication between software and hardware.

Until 2008 the four-legged Sony Aibo robot was used as the common platform. Starting from 2008 the humanoid robot Nao produced by Aldebaran is used. Because of 10 years difference, the Nao can benefit from progress in technology.

But, the restriction of processing power is still a bottle neck, because reactions have to be done in real time, e.g., perception and control calculations should be finished during the cycles given by the frequency of images (30 frames per second). Otherwise, the robot would act on obsolete data. The runtime system uses a special middle ware, the Naoqi.

Of course, biped motion is much more complicated than quadruped motion, and skills like kicking and dribbling are still under development. For the Aibo, the RoboCup teams could provide a walk of about 50cm/second which was about three times faster than the original walk, this was mainly done using Machine Learning.

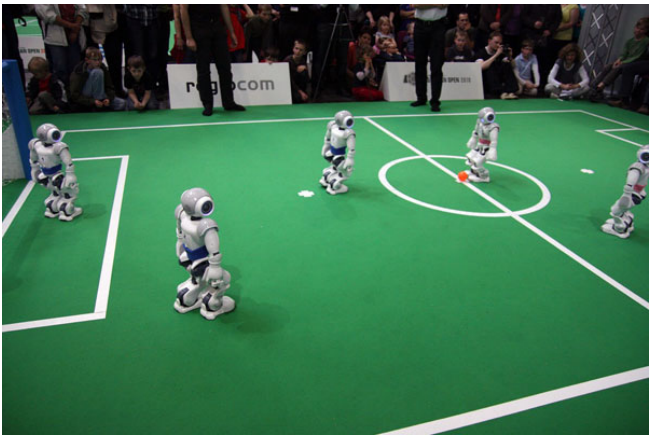

Fig. 2. A 3 vs. 3 game is playing in the Standard Platform League at the German Open 2010 in Magdeburg

2.3 Comparison of Two Leagues

Since Standard Platform League and 3D Simulation League use the same robot — Nao from Aldebaran Robotics, it provides a good opportunity to cooperate between two leagues, but there are also some difference between them. In this subsection, we investigate the difference and consistency of the two platforms.

First of all, the simulated robot in 3D Simulation League has some differences to real Nao robot, see Table 1. In the real Nao, the left hip and right hip are physically connected one motor so they cannot be controlled independently, but there are two motors in the simulated robot. The robot program has to communicate with NaoQi, but the network connection is established between agent program and simulator.

The vision of real Nao is based on CMOS camera, it can receive 640×480 pixels image in 30 FPS. The image processing is not needed in simulation up to now, since the robot receives abstract vision percepts from simulator directly. The vision percepts of each object contains distance and angle relative to the

Table 1. Comparison between real Nao robot and simulated robot in 3D Simulation League

	Standard Platform League	3D Simulation League
Degree of freedom	25	26
Kinematics	the same	the same
Joint control	angle	velocity
Vision	2 cameras	vision information
Accelerometer	3 axises	3 axises
Gyrometer	2 axises	3 axises
Force sensor in each foot	4 force sensitive resistors	6 dimensions force sensor
LEDs	yes	no
Loudspeakers & Microphones	yes	no
Sonars	yes	no
IR transmitter/receiver	yes	no
Platform interface	NaoQi	TCP/IP

camera coordinates. The real robot has to calculate related percepts by image processing. Using percepts, the real robots as well as the simulated robots have to calculate higher level information, e.g., for localization and speed of objects.

The head, body, hands and feet of other robots are given by the vision percepts in simulation. It provides much more information than real images, so it is possible that the robot can better recognize the situation of the environment and perform more complex tasks.

The simulated force sensor in the foots also provides more information than for the real robot, since it provides force vector and force point at the same time. The gyrometer has one axis more than in the real robot.

Besides the robot, the game in different leagues are organized in different ways, see Table 2. The field size and goal width in 3D Simulation League is bigger than for Standard Platform League. Each team has 6 robots in Simulation League while only 3 robots are used in Standard Platform League. The duration of one Standard Platform League game is twice long as simulation game. In comparison to Standard Platform League, the rules of 3D Simulation are closer to human soccer rules.

Table 2. Comparison of games in Standard Platform League and 3D Simulation League

	Standard Platform League	3D Simulation League
Field size	6m×4m	15m×10m
Goal size	1.4m×0.8m	2.1m×0.8m
Number of robots	3	6
Game	without stop	with kick in, goal kick and etc.
Duration	20 min	10 min

3 Implementation

In order to play both in Simulation 3D League and Standard Platform League, the architecture strives to seamlessly integrate simulation system with real robotic hardware, and allows simulated and real architectural components to function seamlessly at the same time. And also it should be able to integrate existing available infrastructures.

3.1 Outlines/Architecture

Since the Humboldt-Universität has a long history in RoboCup, there is a lot of experience and already existing code, especially from the GermanTeam[13] with its module based architecture. In oder to integrate different platforms, our project is divided into two parts: platform independent part and platform specific part. The platform independent part is the Core, which can run in any environment, and all the algorithms are implemented here. The platform specific part contains code which is applied to the particular platform. In the core part, several different modules are implemented under the module based architecture. In this section, we briefly describe our modules.

The recent perception model in 3D simulation league is based on already "preprocessed" information provided by the soccer server. For the real robots, we have developed basic methods as well, which might be useful in the future 3D league. Basic image processing as color classification is performed using a 80×60 pixel grid. Object detection (ball, goals, lines, robots) applies classical methods, e.g., region growing. The camera matrix is computed by the kinematic chain (alternative approaches using constraints are under investigation). Constraints are already used for navigation and world model as well as particle filters. A Kalman filter is applied to generate a local ball model. Global ball positions are communicated between players to coordinate team behavior.

Behavior is executed through a hierarchical state machine, known as XABSL (Extensible Agent Behavior Specification Language) [10]. The behavior code generation is supported by the related editor, debugging, and visualization tools. It allows layered usage of different behaviors, from very low-level motions to high-level cooperative team skills. Each option in the behavior tree decides the running time of it's direct active sub-options in case of being activated itself, though each one takes care of it's own timing. Furthermore, options have some internal state criteria, used as inter-option communication means, mainly utilized by sub-options and their super options. Each option can also have a possible transition condition, through which along with the option's state propagation, the active behavior-switching in any decision level can be safely accomplished. In Section 4 we describe some experiments how behavior can be designed in XABSL in order to be used simultaneously in simulation and on a real robot.

Motions are implemented by keyframe techniques and inverse kinematics with motion planning. Keyframe nets were developed through teaching and hand coding with the help of our motion editor framework. For parameterized dynamic

motions as omnidirectional walks and kicks we use and investigate inverse kinematics and the sensor feedback [12].

Debugging code can be switched on/off during runtime. Debug results are transferred over the network and monitored/visualized using RobotControl, a robot data monitoring and debugging tool, which is implemented in Java to be used on arbitrary computer systems, see Fig. 3. It is a good tool to analysis and debug one robot, but in order to analysis and develop the team behavior of a robot soccer team, we need to connect to all the robots at the same time. A related new tool — TeamControl is under the development now.

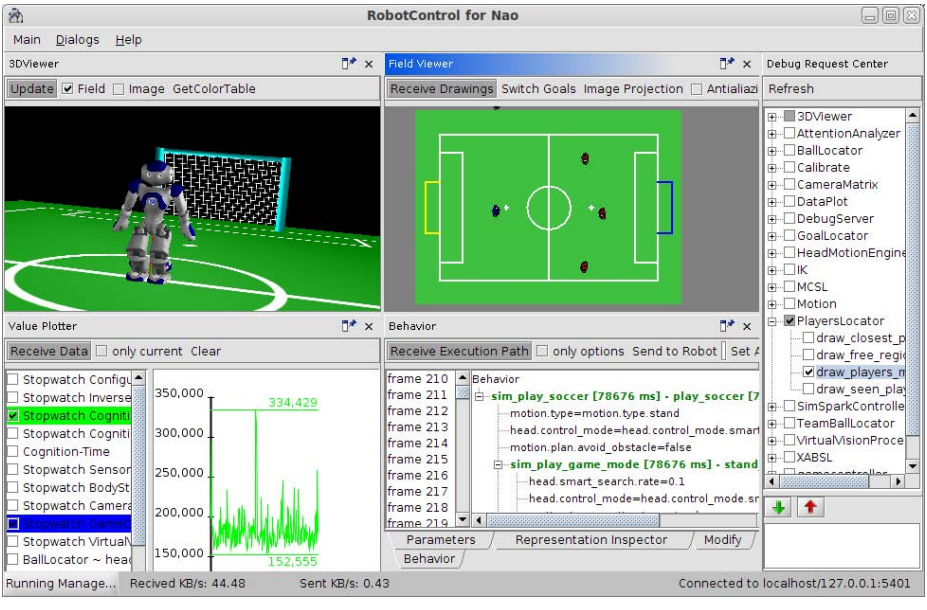

Fig. 3. The RobotControl program contains different dialogs. In this figure, the left top dialog is the 3D viewer, which is used to visualized the current state of robot; the left bottom dialog plots some data; the middle top dialog draws the field view; the middle bottom shows the behavior tree; and the right one is the debug request dialog which can enable/disable debug functionalities.

3.2 Unified Platform Interface

At present, our agent (robot control program) can run in 4 different platforms: real *Nao* robot, *Webots* simulator, *SimSpark* simulator and *log simulator*. An abstraction interface is implemented in the sense - think - act loop. With the unified platform interface(see Fig. 4), the core of our program is independent of the platform which it runs, and all the platform dependent codes are in separate platform implementations.

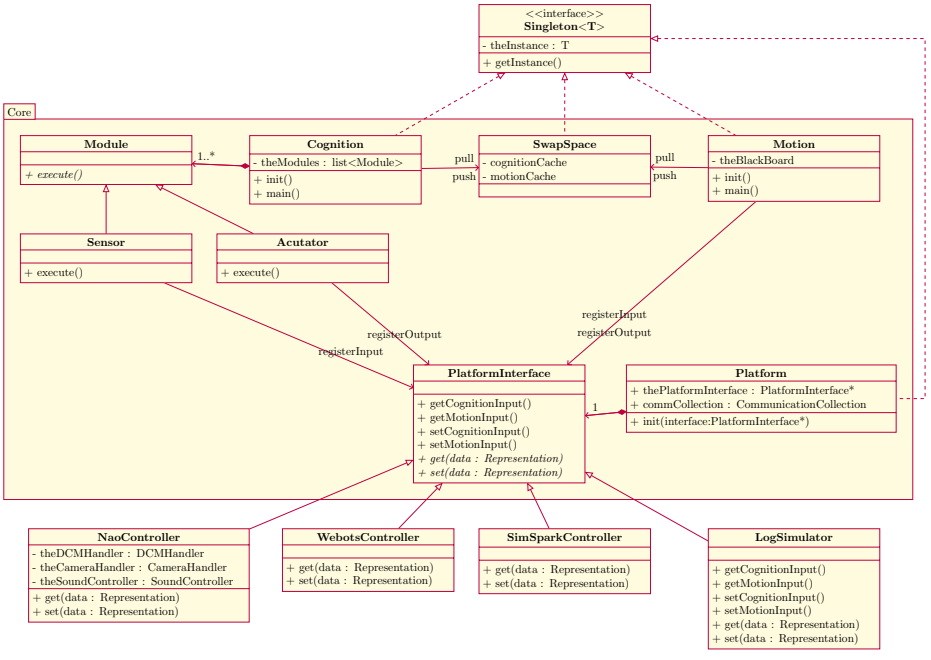

Fig. 4. Framework of unified platform interface, the core of program can run in different platforms with different modules

As mentioned in section 2, there are some difference between simulation and real robot. Comparing to real Nao robot, some devices are missing in the simulations, such as LEDs and sound speaker. In this case, the platform skips unsupported devices, and uses different modules for different platforms. For example, camera (image sensor) is not used in 3D simulation league. Thus, the controller can disable the image processing module, use the virtual *see* sensor of the simulator SimSpark to provide perceptions. Therefore, the core can run on different platforms and enable different modules for different sensors and actuators.

4 Experiments and Results

With the implementation of architecture and modules, our robot program can be executed on different platforms. In order to play both in 3D Simulation League and Standard Platform League, related experiments were performed and investigated, especially for behavior and motion.

The behavior of our robot program is build by XABSL (cf. 3.1). It is relatively easy to describe some behavior from soccer theory. For example, the role changing can be described by 4 basic states, see Fig. 5. But the decision of changing between states depends heavily on the characteristics of the robot and the environment, i.e. the conditions for decisions are different in different leagues. Currently, some parameters are chosen specially for different leagues.

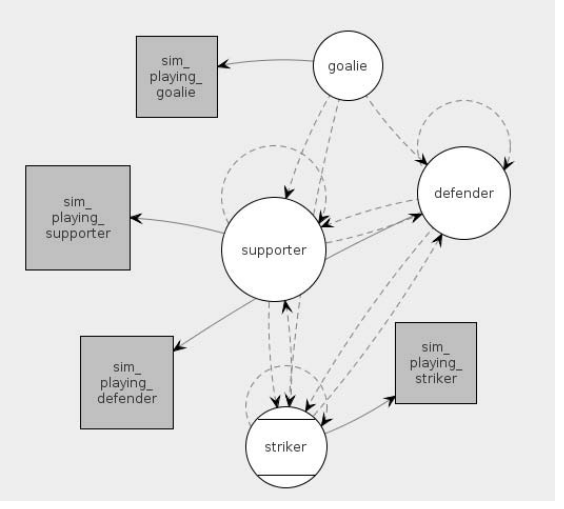

Fig. 5. The role changing behavior described by XABSL

The better alternative is creating some symbols by lower level control modules, which makes the decision processes more general. For example, the motion module provides parameters of motion skills like the walking speed. Then the behavior module can base its decisions on real values. We have successfully used related behaviors to play in both leagues, i.e. for skills like go to ball and kick, and dribble, respectively.

Our motions are implemented by key frame techniques and parameterized dynamic motions. For key frame motion, we adjust key frames for the different platforms. For parameterized dynamic motions, evolutionary approaches are used to optimize the parameters. For example, our biped walking is implemented for the real robot with some parameters which are tested on the real Nao. Basically, it also works in simulation, but it is not fast enough to play against other teams. We use evolution algorithm as well to optimize the 18 parameters of our walking in simulation. After 50 generations with 100 individuals in each generation, our robot runs at the same speed as the fastest team in simulation league.

With these experiments, our team successfully participated RoboCup German Open 2010, and won the 4th place in Standard Platform League and 1st place in 3D simulation league. It was the first time that the same source code played in two leagues: both simulation and real robot.

5 Discussion

We have developed a framework for running robots both in real and simulated settings. A main challenge is the usage of unified code as much as possible. Now, the performance of a control program in the simulation can be compared with on

a real robot. Differences occur systematically from differences between sensors and actuators.

Unrealistic assumptions about the statistics of the sensor input will result in differences on the higher level control. Sensory inputs in real robots are very noisy and biased data. The difference between the simulated sensory input and the sensory input that comes from a real world humanoid robot system can be directly recorded and compared. In this way we can get accurate statistics and can integrate the results into the simulated sensory environment. It is possible to establish feedback from the reality to the simulation league, and the related statistics can lead to stepwise improvement of the 3D simulator.

As another common experience, there are big gaps between simulation and reality in basic physics with consequences for low level skills in motions. An challenging question is the transferring of learned skilled from simulation to reality. We investigate the relationships and possibilities for the transferring of methods and code. Again, consequences can lead to better simulation tools, especially in the 3D Simulation League.

At the higher level, the aim is to use the same tactics in simulation and in reality. Simulation still provides better opportunities to develop and test coordination strategies (e.g., by larger fields with more players). Here, simulation will be the driving force for better play.

So far, we have developed walking gaits through evolution techniques in a simulated environment [7,6]. Reinforcement Learning was used for the development of dribbling skills in the 2D simulation [14], while Case Based Reasoning was used for strategic behavior [4,2]. BDI-techniques have been investigated for behavior control, e.g., in [1,3].

6 Conclusion

We investigated Standard Platform League and Simulation League for getting simulation and real robot closer. Our unified interface and modular architecture are presented. As a result, our team — Nao Team Humboldt uses a common Core of our program for both platforms. We successfully participated RoboCup German Open 2010, and won the 4th place in Standard Platform League and 1st place in 3D simulation league. It was the first time that the same source code played in two leagues: both simulation and real robot.

The key point to success is narrowing the gap between simulation and reality. There are some researchers who have already tried to achieve this, but there are only few successful results so far. At first, the relations (differences and similarities) between reality and simulations should be better understood. This will help to develop more realistic simulations. Simulations than can help to develop methods and code for real robots in an easier way. Our plan is to analyze data from sensors/actuators in simulation and from real robots at first and then to apply machine learning methods to improve the available model or to build a good implicit model of the real robot from the data.

Acknowledgments

The authors would like to thank other members of Nao Team Humboldt.

References

1. Berger, R.: Die Doppelpass-Architektur. Verhaltenssteuerung autonomer Agenten in dynamischen Umgebungen. Diploma thesis, Humboldt-Universität zu Berlin, Institut für Informatik (2006) (in German)
2. Berger, R., Lämmel, G.: Exploiting Past Experience. Case-Based Decision Support for Soccer Agents. In: Hertzberg, J., Beetz, M., Englert, R. (eds.) KI 2007. LNCS (LNAI), vol. 4667, pp. 440–443. Springer, Heidelberg (2007)
3. Burkhard, H.D.: Programming Bounded Rationality. In: Proceedings of the International Workshop on Monitoring, Security, and Rescue Techniques in Multiagent Systems (MSRAS 2004), pp. 347–362. Springer, Heidelberg (2005)
4. Burkhard, H.-D., Berger, R.: Cases in robotic soccer. In: Weber, R.O., Richter, M.M. (eds.) ICCBR 2007. LNCS (LNAI), vol. 4626, pp. 1–15. Springer, Heidelberg (2007)
5. Dylla, F., Ferrein, A., Lakemeyer, G., Murray, J., Obst, O., Röfer, T., Stolzenburg, F., Visser, U., Wagner, T.: Towards a league-independent qualitative soccer theory for robocup. In: Nardi, D., Riedmiller, M., Sammut, C., Santos-Victor, J. (eds.) RoboCup 2004. LNCS (LNAI), vol. 3276, pp. 29–40. Springer, Heidelberg (2005)
6. Hein, D.: Simloid – Evolution of Biped Walking Using Physical Simulation. Diploma thesis, Humboldt-Universität zu Berlin, Institut für Informatik (2007)
7. Hein, D., Hild, M., Berger, R.: Evolution of biped walking using neural oscillators and physical simulation. In: Visser, U., Ribeiro, F., Ohashi, T., Dellaert, F. (eds.) RoboCup 2007: Robot Soccer World Cup XI. LNCS (LNAI), vol. 5001, pp. 433–440. Springer, Heidelberg (2008)
8. Kitano, H., Asada, M., Kuniyoshi, Y., Noda, I., Osawa, E.: RoboCup: The robot world cup initiative. In: Johnson, W.L., Hayes-Roth, B. (eds.) Proceedings of the First International Conference on Autonomous Agents (Agents 1997), pp. 340–347. ACM Press, New York (5-8, 1997), `citeseer.ist.psu.edu/kitano95robocup.html`
9. Kögler, M., Obst, O.: Simulation league: The next generation. In: Polani, D., Browning, B., Bonarini, A., Yoshida, K. (eds.) RoboCup 2003. LNCS (LNAI), vol. 3020, pp. 458–469. Springer, Heidelberg (2004)
10. Lötzsch, M., Risler, M., Jüngel, M.: Xabsl - a pragmatic approach to behavior engineering. In: Proceedings of IEEE/RSJ International Conference of Intelligent Robots and Systems (IROS), Beijing, China, October 9-15, pp. 5124–5129 (2006)
11. Mayer, N.M., Boedecker, J., da Silva Guerra I, R., Obst, O., Asada, M.: 3d2real: Simulation league finals in real robots. In: Lakemeyer, G., Sklar, E., Sorrenti, D.G., Takahashi, T. (eds.) RoboCup 2006: Robot Soccer World Cup X. LNCS (LNAI), vol. 4434, pp. 25–34. Springer, Heidelberg (2007)
12. Mellmann, H., Xu, Y.: Adaptive motion control with visual feedback for a humanoid robot. In: Proceedings of the 2010 IEEE/RSJ International Conference on Intelligent Robots and Systems (to appear, 2010)

13. Röfer, T., Brose, J., Göhring, D., Jüngel, M., Laue, T., Risler, M.: GermanTeam 2007 - The German national RoboCup team. In: RoboCup 2007: Robot Soccer World Cup XI Preproceedings. RoboCup Federation (2007)
14. Uc-Cetina, V.: Reinforcement Learning in Continuous State and Action Spaces. Ph.D. thesis, Humboldt-Universiät zu Berlin (2009)
15. Wang, J., Balakirsky, S.: USARSim — A Game-based Simulation of the NIST Reference Arenas, University of Pittsburg (May 2005), `http://sourceforge.net/projects/usarsim/`