GermanTeam 2004

The German National RoboCup Team

Thomas Röfer[1], Ronnie Brunn[2], Ingo Dahm[3], Matthias Hebbel[3], Jan Hoffmann[4], Matthias Jüngel[4], Tim Laue[1], Martin Lötzsch[4], Walter Nistico[3], and Michael Spranger[4]

[1] Bremer Institut für Sichere Systeme, Technologie-Zentrum Informatik, FB 3, Universität Bremen, Postfach 330 440, 28334 Bremen, Germany
[2] Fachgebiet Simulation und Systemoptimierung, Fachbereich Informatik, Technische Universität Darmstadt, Hochschulstraße 10, 64289 Darmstadt, Germany
[3] Lehrstuhl für Datenverarbeitungssysteme, FB Elektrotechnik und Informationstechnik, University of Dortmund, Otto-Hahn-Strasse 4, 44221 Dortmund, Germany
[4] Institut für Informatik, LFG Künstliche Intelligenz, Humboldt-Universität zu Berlin, Rudower Chaussee 25, 12489 Berlin, Germany

http://www.robocup.de/germanteam
germanteam@informatik.hu-berlin.de

1 Introduction

The GermanTeam participates as a national team in the Sony Legged Robot League. It currently consists of students and researchers from the following four universities: the Humboldt-Universität zu Berlin, the Universität Bremen, the Technische Universität Darmstadt, and the Universität Dortmund. The members of the GermanTeam participate as individual teams in contests such as the RoboCup German Open, but jointly line up as a national team for the international RoboCup World Cup. To support this cooperation and concurrency, the GermanTeam introduced an architecture that provides mechanisms for parallel development [1]. The entire information processing and control of the robot is divided into *modules* (cf. fig. 1) carrying out specific tasks using well-defined interfaces. For each module, many different *solutions* can be developed which can be switched at runtime. This approach allows for easily comparing and benchmarking the solutions developed for the various tasks. It is used since the fall of 2001 and has been proven to be a very useful tool for development. In 2004, the only two newcomers in the Sony Four-Legged Robot League (Dutch Aibo Team and Hamburg Dogbots) will also be based on this architecture.

This paper gives an overview on the work done by the four sub-teams of the GermanTeam (Aibo Team Humboldt, Darmstadt Dribbling Dackels, Bremen Byters, and Microsoft Hellhounds) in the past year that is currently combined to form the code of the GermanTeam 2004. Further information can be found in this year's contributions of members of the GermanTeam to the RoboCup book: they deal with automatic color calibration [2], object recognition [3], obstacle avoidance [4] (cf. Sect. 3), collision detection [5] (cf. Sect. 4), qualitative world modeling [6, 7], behavior modeling [8], and gait optimization [9] (cf. Sect. 7).

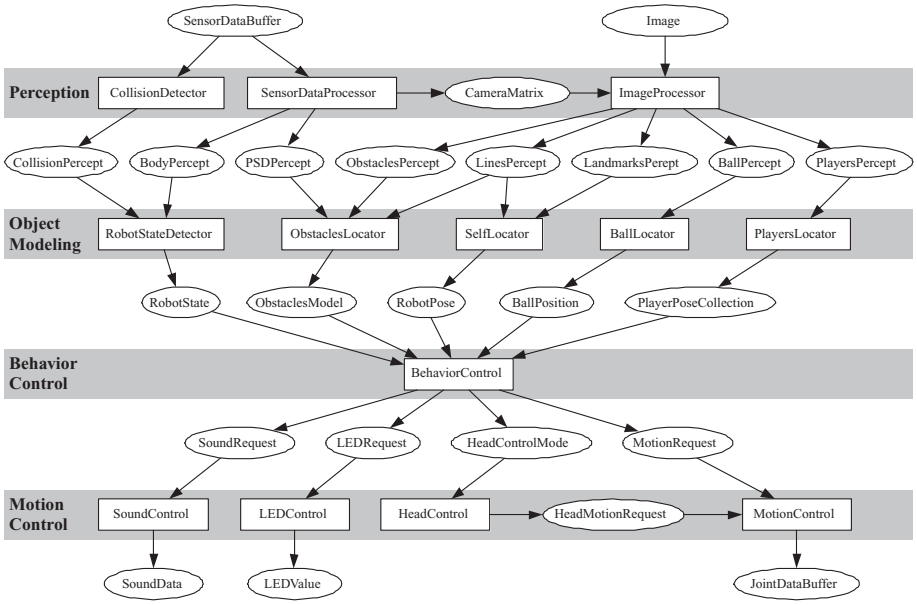

Fig. 1. Information processing in the robots of the GermanTeam. Boxes denote *modules*, ellipses denote the *representations* that are needed to exchange information between the modules.

2 Perception

Image Processing. Porting the code to work on the ERS7 robot has involved several challenges: the new camera has a lower sensitivity, requiring the use of a high gain to compensate for the underexposed pictures; worse, the large radiometric distortion exhibited is a serious issue, with pronounced vignetting at the image corners, which appear darker and characterized by a strong blue cast.

The first problem has been addressed by developing a "real time" implementation of a structure-preserving noise reduction filter based on the S. U. S. A. N. principle [10], which sacrifices certain features like Gaussian brightness, spatial and frequency response to make use of only fast operations as look-up, shifting and masking.

To correct the second problem, the distortion has been modeled as the product of two polynomial factors, one based on the distance of the pixel from the center of the image (radial factor), the other dependent on the actual value of the pixel spectrum to be corrected (brightness factor), as the deviation from the values at the center of the image has been observed to be dependent on the actual brightness of the reference.

To derive the coefficients for the correction model, a log file of images of uniform colors such as blue, yellow, white, recorded from the robot camera is

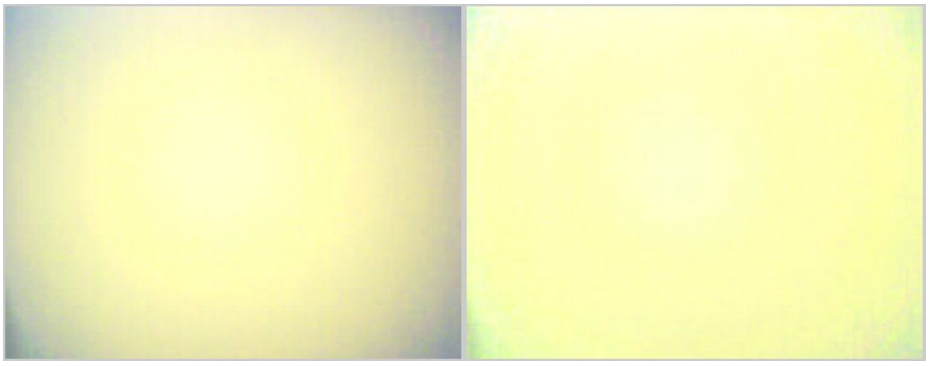

Fig. 2. Left: the image of a white card captured by the robot camera. Right: the color corrected image

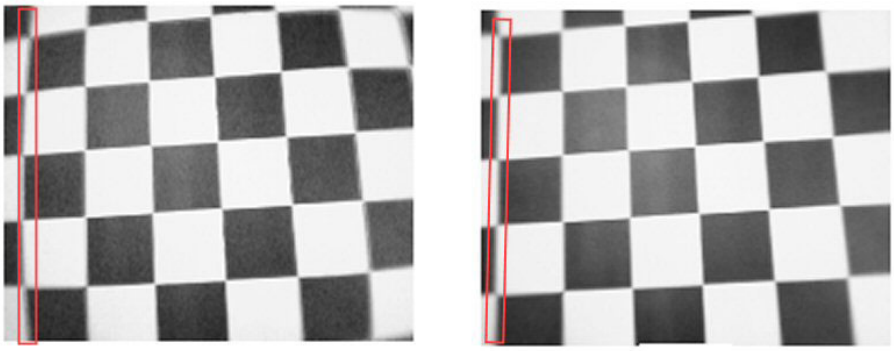

Fig. 3. The red selection highlights the effect of the distortion (left) and the result of the correction (right).

processed to estimate the reference value for all the spectra for each color, and the sum of the squared differences of each pixel in the images from the references is minimized through simulated annealing.

At last, the barrel distortion induced by the lens geometry has been modeled with the help of a Matlab toolbox from [11] analyzing images of a checkerboard taken under different angles; the correction makes use of a 4th order radial-only model.

Object Recognition. Until now the algorithms analyzing the image highly rely on the use of manually calibrated color tables. Some efforts have been made to automatically adapt to different lighting conditions completely without color tables or at least to generate the color table automatically ([12, 13, 2]). As these methods in some situations perform not as precise as a manual calibrated vision system they were not used for the games. However some specialized algorithms,

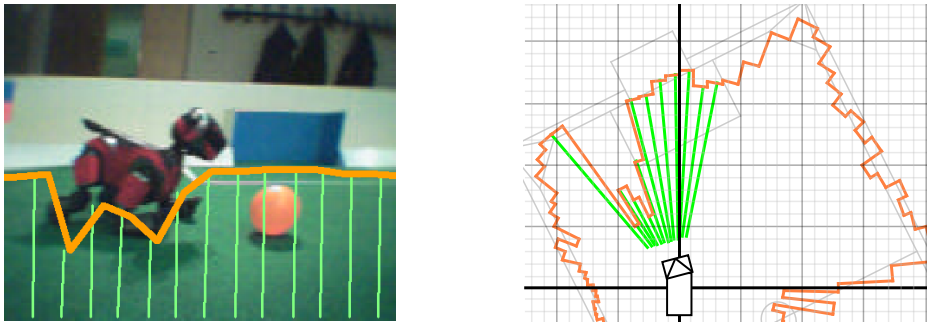

Fig. 4. The obstacle model as seen from above and projected into the camera image.

such as the detection of the ball could be modified, to be more independent from the color tables.

Instead of detecting the ball only by scanning for pixels which are classified as orange by the color table, the modified approach searches for the edges of the ball, by calculating the alikeness of the color compared to the prototype of a perfect orange and scanning for a sharp drop of this value. During the manual color calibration colors which can be seen on the ball but are not strictly orange, i.e. highlights or shadows, are no longer needed to be classified as orange in the color table, which proved to be of great benefit preventing the detection of nonexistent balls, e.g. within the yellow goal or the red jerseys of other robots.

Combined with other optimizations in image processing, like the utilization of the high resolution images, and especially more sophisticated ball detection this approach produced more accurate results. Even at a great distance and being partially concealed by other robots or cut off by image borders the ball could be detected correctly, i.e. its radius and center point were measured with a higher accuracy improving the subsequent ball modeling.

3 Obstacle Avoidance

A vision based system is used for obstacle avoidance. An early version of it was used successfully in the RoboCup 2003 games and in the RoboCup 2003 obstacle avoidance challenge where it won 1st place. The system enables the robot to detect unknown obstacles and reliably avoid them while advancing toward a target. It uses monocular vision data with a limited field of view. Obstacles are detected on a level surface of known color(s). A radial model is constructed from the detected obstacles giving the robot a representation of its surroundings that integrates both current and recent vision information 4. Sectors of the model currently outside the current field of view of the robot are updated using odometry. The modeling of obstacles bears strong resemblance to the method called *visual sonar* which was developed at the same time at CMU [14].

The obstacle model was improved by not only storing obstacles but also the type of obstacle that was detected by vision (i.e. field borders, goals, players). This information can be used to determine where the robot should play the ball (e.g. in the direction of a player of the own team).

Furthermore the algorithm was greatly simplified yielding a further performance improvement. A detailed description can be found in [4].

4 Collision Detection

In the case that obstacle avoidance fails or is impossible (e.g. when walking sideways or while the robot is turning), detecting collisions is advantageous. Collision detection can be used to improve localization. If a collision occurs or the robot cannot move freely, odometry will not correctly reflect the actual motion of the robot anymore. Detecting such incidents can help determining the quality of odometry. Furthermore, detecting collision can used to allow the robot to trigger motions to free it from other robots charging at it.

Collision detection was implemented based on the comparison of sensor readings (actual motion) to actuator commands (intended motion). Ways of detecting such incidents were investigated using just the sensor readings from the servo motors of the robot's legs. It was found that comparison of motor commands and actual movement (as sensed by the servo's position sensor) allowed the robot to reliably detect collisions and obstructions. Minor modifications to make the system more robust enabled it to be used in the RoboCup domain, allowing the system to cope with arbitrary movements and accelerations apparent in this highly dynamic environment. Strong emphasis was put on keeping the process of calibration for different robot gaits simple and manageable. The algorithms where originally developed for the Sony ERS-210 where it was possible to reliably detect collisions [5]. The algorithm was ported to work with the new Aibo ERS-7 which proved to be a difficult task because of the largely different hardware.

5 Behavior Control

For behavior engineering, the GermanTeam continued to use the *Extensible Agent Behavior Specification Language* (XABSL) [15, 16]. Only minor changes were made in the *XABSL* language and system itself. Also the high level behaviors were only slightly changed from 2003. Much work was done on smoother ball handling behaviors including dribbling, kick selection, and navigation, as described below.

Ball Handling. The main goal was to develop a behavior that handles the ball as fast as possible. First a behavior was implemented that plays the ball without any kicks—the robot just runs into the ball. To dribble the ball to the left or to the right the robot performs a short turn when it approaches the ball. To bring the ball behind itself, the robot catches the ball with the head, turns around and

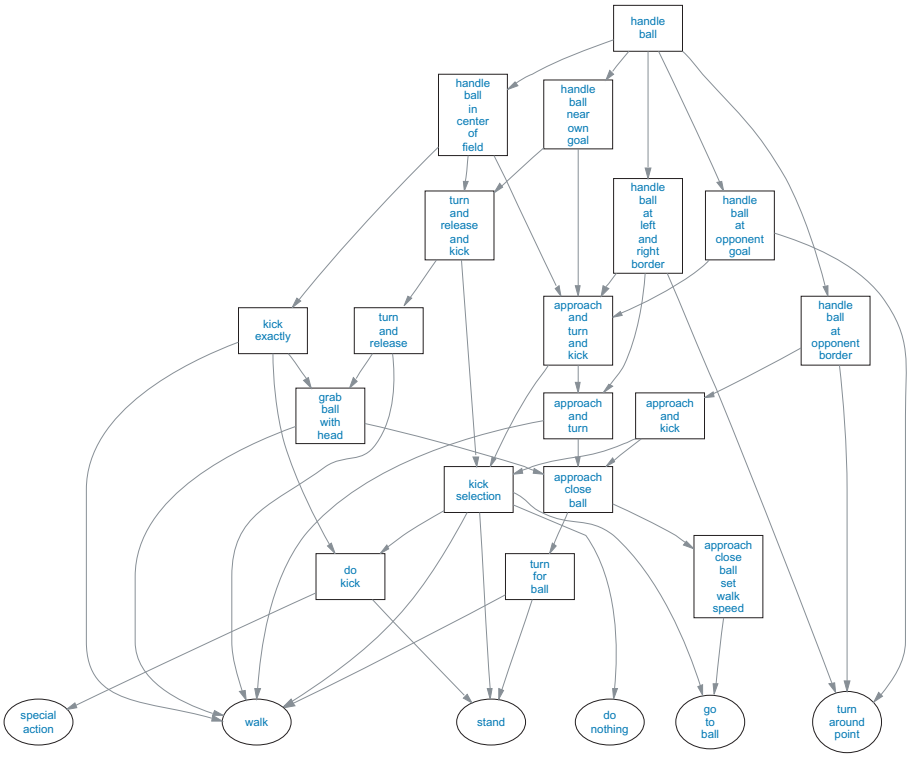

Fig. 5. The ball handling part of the *XABSL* option graph. More complex options (boxes) are each combined from simpler options and basic behaviors (ellipses).

releases the ball at an appropriate angle by lifting its head. In a second step kicks were added to the robot's behavior. If the ball is by chance in a good position while the robot approaches it, a convenient kick is performed. The fundamental idea is that the behavior can react in the majority of situations that might occur (relation between the robot and the ball) and does not try to cause situations where it has a reaction ready.

Kick Selection. Kicking fast and precisely is crucial when playing robot soccer. Thus, about 50 different kicks were developed, suitable for almost all situations that can happen during a match. Thus a large amount of specialized kicks requires an evaluation function to select which kick should be used in a certain situation. Developing such a function by hand and fine-tune its parameters is a time consuming process. Instead, a semi-autonomous teach-in mechanism was developed. Thereto, a robot stands on the playing field and kicks the ball several times. Meanwhile, the starting position and the final position of the ball are measured relative to the robot. The distribution of the final positions is es-

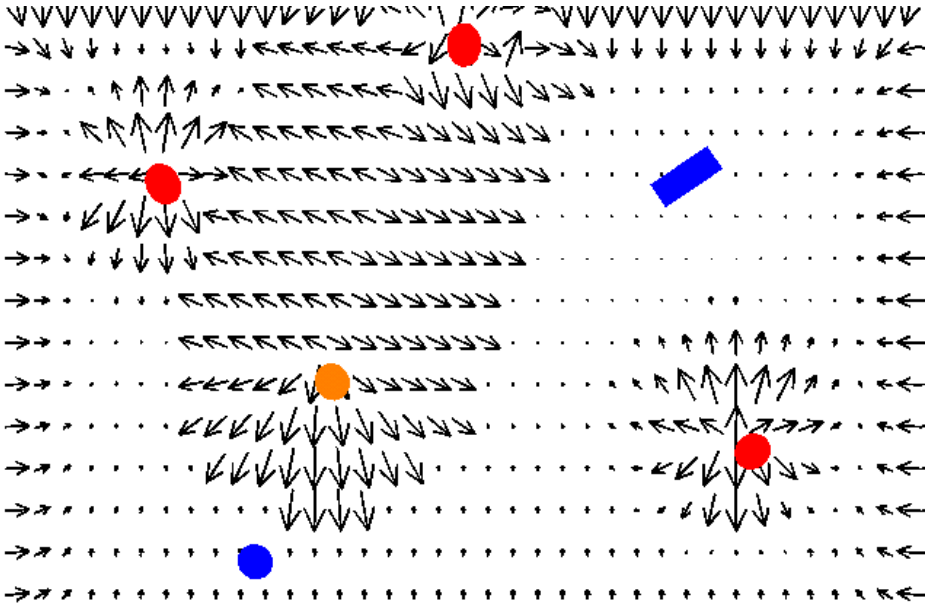

Fig. 6. A potential field describing the behavior of an offensive supporter (blue rectangle). While a teammate (blue circle) approaches the ball, the robot moves to an obstacle-free region near the opponent goal and also avoids blocking a possible kick by its teammate.

timated, and its variance is computed. Thus, it is possible to estimate an angle and distance to the final position depending of the starting position of the ball in each situation of the game.

Potential Fields for Navigation. Artificial potential fields, originally developed by [17], are a quite popular approach in robot motion planning, because of their capability to act in continuous domains in real-time. By assigning repulsive force fields to obstacles and an attractive force field to the desired destination, a robot can follow a collision-free path via the computation of a motion vector from the superposed force fields. Instead of the *Continuous Basic Behaviors*, which have previously been used by the German Team for this kind of robot navigation [18], the potential field architecture by [8] will be used for several navigation tasks during the upcoming competitions. Figure 6 shows an example.

The approach combines a variety of existing techniques in a behavior-based architecture using the principle of competing independent behaviors [19]. The specification of the behaviors and the environment is, quite similar to XABSL, based on external configuration files.

Dynamic Team Tactics. The positioning of the robots and their cooperation is the basis of efficient team-play. Therefore, it is suggested extending the XABSL-

behavior by a meta-layer that helps to represent the dynamics of the game and environment – the Dynamic Team Tactics (DTT). The DTT is based on a communicated world model. Thus, each robot knows about the position and information of all team members. First, all robots estimate what task would be appropriate to solve the actual situation. Then, they compute which task is the most preferable (highest chance of success – COS) for their own to execute. Finally, the robot executes the task with the highest COS-value.

This approach leads to the problem of evaluating the position and prospect of success for every robot and each task. By now, these values are tuned by hand. In the next year, it will be investigated in adjusting the values automatically by a training scheme.

6 Active Vision

In the games in Fukuoka, active vision was used to actively search for landmarks when chasing the ball. With the introduction of the obstacle model, the approach was no longer pursued since it seemed to make more sense to scan the area around the robot. Now, it is returned to an approach that allows for directed vision while also gathering information needed for the obstacle model.

If the robot is looking at the ball, its gaze direction is optimized to also look at the next landmarks while keeping the ball within its field of view. The beacons on the field and the field markings are considered landmarks. If it is impossible to see the ball and landmarks at the same time, the position of the closest landmark is calculated and the robot quickly verifies its localization by looking at such a landmark. (In the case of a badly localized robot, such a quick look usually yields enough information for the localization to recover.) The need for wide scanning motion of the head is lessened by making sure that the robot is walking forward most of the time. This approach enables the robot to more closely track the ball.

7 Motion Modeling and Gait Evolution

To generate the walk motions of the robot, the GermanTeam uses a walking engine based on inverse kinematic calculations. The walking engine is very flexible since the resulting walk depends on a huge number of parameters. With the current implementation of the walking engine, it is possible to change the parameter set during runtime, such that the appropriate walk can be observed immediately. In general, different walking parameters result in different walking styles with different speeds. The task to find a fast and effective parameter set describing the walk becomes more and more difficult with an increasing number of parameters. Finding the fastest possible walk using a walking engine with n parameters means to find the representation of the fastest walk in an n-dimensional search space. For a large number n this is not feasible by trying different parameter combinations by hand. Two different approaches to optimize the gait parameters were used in the GermanTeam:

Localization-Based Fitness. A basic approach to find a parameter set resulting in a fast walking pattern is to use a simple $(1 + 1)$ evolution strategy [20] for the optimization process. An individual is represented by a set of walking parameters. Its fitness is discovered by measuring the corresponding speed of the robot while walking on the field. This is done in the following way: At first the robot walks to a starting position, after arrival it walks with the maximum speed for a defined time (we used 10 seconds) on the field. After that time the robot stops, localizes itself and calculates the walked distance and thereby the speed of the walk. To reduce measurement errors, the robot measures the walking speed three times and uses the median of the measured speeds. Due to the reliable self-localization of the robots the measurement of speed and therefore the complete evolution process can be done fully autonomously. Therefore, it is possible to adapt the walking parameter set to the actual environmental conditions (e.g. carpet).

Odometry-Based Fitness. The other approach to gait optimization employs a probabilistic evolution scheme [9]. It separates the typical crossover-step of the evolutionary algorithm into an interpolating step and an extrapolating step, which allows for solving optimization problems with a small population, which is an essential for robotics applications. In contrast to other approaches, real odometry is used to assess the quality of a gait. The motion of the robot is estimated from the trajectories of the rear legs while they have contact to the ground. The main advantage is that gaits can be assessed very quickly (every 5 seconds a new one), so learning is very fast. The only drawback of this approach is that it can only learn gaits in which the ground contact sensors of the rear feet touch the ground. With this approach, a maximum forward speed of 40 cm/sec and a turning speed of 195°/sec were reached.

8 RobotControl

In recent years, the GermanTeam has developed a big variety of tools that support the development process. Amongst them, *RobotControl* is a very general debug tool for the work with the Aibos. This extensive and complex debugging tool (cf. Fig. 7) evolved over the years together with the software architecture of the GermanTeam. First, it is a viewer for almost all intermediate steps of information processing on the robots. It can connect via WLAN to up to 8 robots and display representations, internal data from modules, and even single drawings from within the algorithms. Second, the highly modular structure allows the developers to easily plug in graphical user interfaces for testing single modules separately. Third, it includes the simulator *SimRobot* which simulates up to 8 Aibos for testing algorithms offline. For that, the complete source code running on the robots is compiled and linked into *RobotControl*. At last, it hosts a variety of general tools, e.g. for color calibration or walk evolution.

From 2003, the modularization of the program was clarified, a possibility for direct TCP communication with the Aibos was implemented, and many new

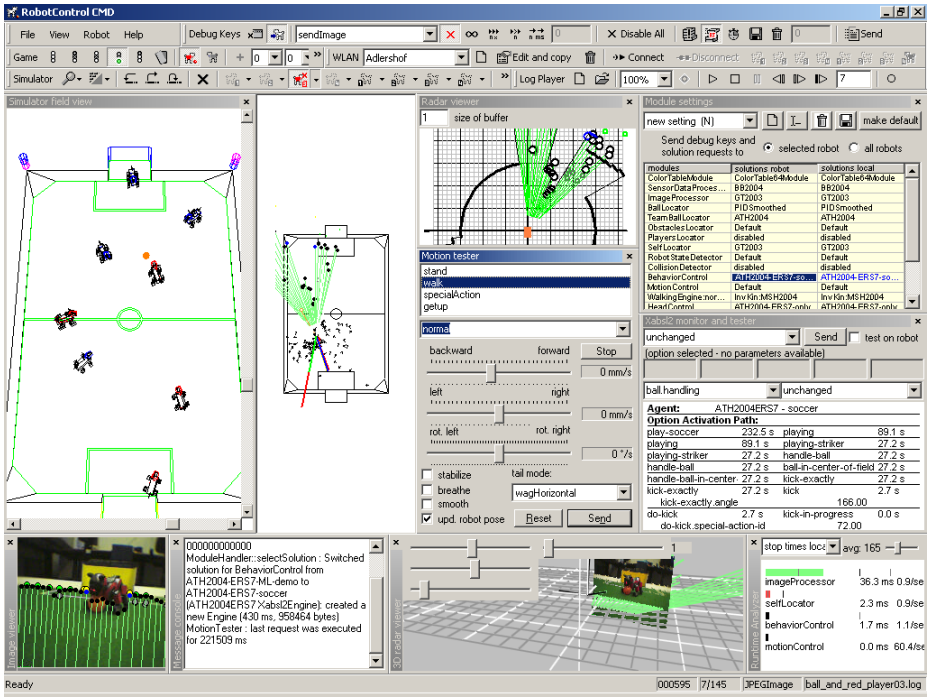

Fig. 7. A screen shot of one of the many possible configurations of *RobotControl*.

graphical interfaces were added to the application. In addition, a new, OpenGL-based version of *SimRobot* (cf. Fig. 8) was developed that has not been integrated yet in RobotControl. This new version is currently enhanced by the ability to simulate the physics of the robots and the environment.

9 Conclusion and Future Work

The GermanTeam has spent a lot of time on porting the code to the ERS-7. The new camera required some calibration to compensate for the color distortion. Collision detection was developed and the behavior was fine tuned. Many efforts have been made within the GermanTeam to generate new gaits for the new Aibo ERS-7. The most successful one was found using genetic optimization and achieved very high speeds, another is also quite fast and still allows performing omni-directional motion. Since these gaits are highly optimized for a specific task, it is challenging to maintaining good overall performance and predictability. This will be achieved by interpolating between different gaits and by automated calibration of odometry. The calibration process will measure odometry for all possible speeds and combinations of motions yielding highly accurate, predictable robot locomotion.

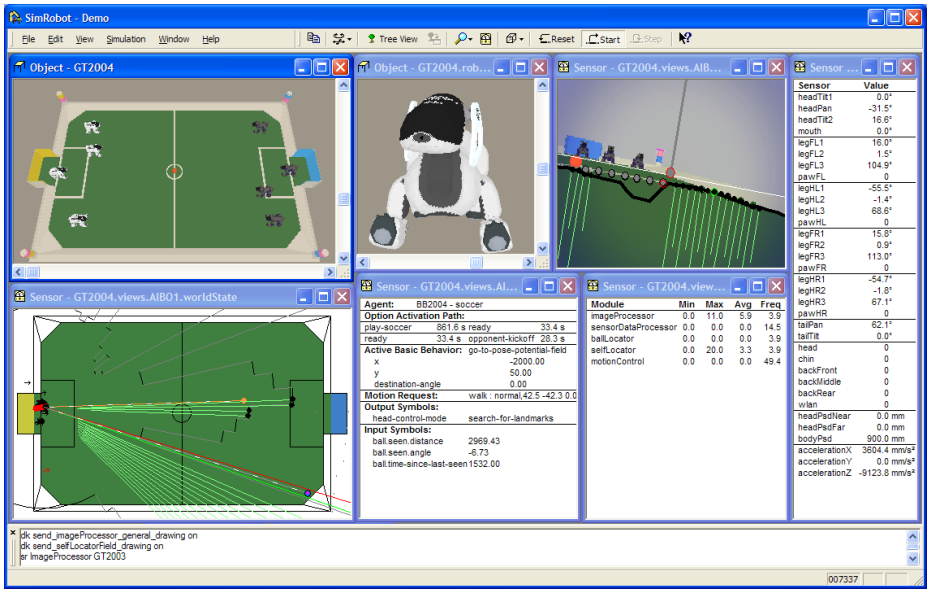

Fig. 8. The new OpenGL-based simulator SimRobot XP.

References

1. T. Röfer, "An architecture for a national robocup team.," in *RoboCup 2002 Robot Soccer World Cup VI, Gal A. Kaminka, Pedro U. Lima, Raul Rojas (Eds.)*, no. 2752 in Lecture Notes in Artificial Intelligence, pp. 417–425, Springer, 2003.
2. M. Jüngel, "Using Layered Color Precision for a Self-Calibrating Vision System," in *8th International Workshop on RoboCup 2004 (Robot World Cup Soccer Games and Conferences)*, Lecture Notes in Artificial Intelligence, Springer, 2005. to appear.
3. D. Wilking and T. Röfer, "Object Recognition Using Decision Tree Learning," in *8th International Workshop on RoboCup 2004 (Robot World Cup Soccer Games and Conferences)*, Lecture Notes in Artificial Intelligence, Springer, 2005. to appear.
4. J. Hoffmann, M. Jüngel, and M. Lötzsch, "A Vision Based System for Goal-Directed Obstacle Avoidance used in the RC03 Obstacle Avoidance Challenge," in *8th International Workshop on RoboCup 2004 (Robot World Cup Soccer Games and Conferences)*, Lecture Notes in Artificial Intelligence, Springer, 2005. to appear.
5. J. Hoffmann and D. Göhring, "Sensor-Actuator-Comparison as a Basis for Collision Detection for a Quadruped Robot," in *8th International Workshop on RoboCup 2004 (Robot World Cup Soccer Games and Conferences)*, Lecture Notes in Artificial Intelligence, Springer, 2005. to appear.
6. F. Dylla, A. Ferrein, G. Lakemeyer, J. Murray, O. Obst, T. Röfer, F. Stolzenburg, U. Visser, and T. Wagner, "Towards a League-Independent Qualitative Soccer Theory for RoboCup," in *8th International Workshop on RoboCup 2004 (Robot*

World Cup Soccer Games and Conferences), Lecture Notes in Artificial Intelligence, Springer, 2005. to appear.
7. T. Wagner and K. Hübner, "An Egocentric Qualitative Spatial Knowledge Representation Based on Ordering Information," in *8th International Workshop on RoboCup 2004 (Robot World Cup Soccer Games and Conferences)*, Lecture Notes in Artificial Intelligence, Springer, 2005. to appear.
8. T. Laue and T. Röfer, "A Behavior Architecture for Autonomous Mobile Robots Based on Potential Fields," in *8th International Workshop on RoboCup 2004 (Robot World Cup Soccer Games and Conferences)*, Lecture Notes in Artificial Intelligence, Springer, 2005. to appear.
9. T. Röfer, "Evolutionary Gait-Optimization Using a Fitness Function Based on Proprioception," in *8th International Workshop on RoboCup 2004 (Robot World Cup Soccer Games and Conferences)*, Lecture Notes in Artificial Intelligence, Springer, 2005. to appear.
10. S. M. Smith and J. M. Brady, "SUSAN - A new approach to low level image processing," *International Journal of Computer Vision*, vol. 23, no. 1, pp. 45–78, 1997.
11. J. Y. Bouguet, "Camera calibration toolbox for matlab." Website: http://www.vision.caltech.edu/bouguetj/calib_doc/.
12. M. Jüngel, "A vision system for RoboCup: Diploma thesis," 2004.
13. M. Jüngel, J. Hoffmann, and M. Lötzsch, "A real-time auto-adjusting vision system for robotic soccer," in *7th International Workshop on RoboCup 2003 (Robot World Cup Soccer Games and Conferences)*, Lecture Notes in Artificial Intelligence, Springer, 2004.
14. S. Lenser and M. Veloso, "Visual Sonar: Fast Obstacle Avoidance Using Monocular Vision," in *Proceedings of IROS'03*, 2003.
15. M. Lötzsch, J. Bach, H.-D. Burkhard, and M. Jüngel, "Designing agent behavior with the extensible agent behavior specification language XABSL," in *7th International Workshop on RoboCup 2003 (Robot World Cup Soccer Games and Conferences)*, Lecture Notes in Artificial Intelligence, Springer, 2004. to appear.
16. M. Lötzsch, "XABSL web site," 2003. http://www.ki.informatik.hu-berlin.de/XABSL.
17. O. Khatib, "Real-Time Obstacle Avoidance for Manipulators and Mobile Robots," *The International Journal of Robotics Research*, vol. 5, no. 1, 1986.
18. T. Röfer, H.-D. Burkhard, U. Düffert, J. Hoffmann, D. Göhring, M. Jüngel, M. Lötzsch, O. v. Stryk, R. Brunn, M. Kallnik, M. Kunz, S. Petters, M. Risler, M. Stelzer, I. Dahm, M. Wachter, K. Engel, A. Osterhues, C. Schumann, and J. Ziegler, "GermanTeam RoboCup 2003," tech. rep., 2003. Available online: http://www.robocup.de/germanteam/GT2003.pdf.
19. R. C. Arkin, *Behavior-Based Robotics*. The MIT Press, 1998.
20. H.-G. Beyer and H.-P. Schwefel, "Evolution strategies – A comprehensive introduction," *Natural Computing*, vol. 1, no. 1, pp. 3–52, 2002.