Reliable Self-Localization, Multirobot Sensor Integration, Accurate Path-Planning and Basic Soccer Skills: Playing an Effective Game of Robotic Soccer*

J.-S. Gutmann, W. Hatzack, I. Herrmann, B. Nebel, F. Rittinger, A. Topor, and T. Weigel
Albert-Ludwigs-Universität Freiburg, Institut für Informatik
Am Flughafen 17, D-79110 Freiburg, Germany

Abstract

Robotic soccer is a challenging research domain because problems in robotics, artificial intelligence, multi-agent systems and real-time reasoning have to be solved in order to create a successful team of robotic soccer players. In this paper, we describe the key components of the CS Freiburg team. We focus on the self-localization and object recognition method based on using laser range finders and the integration of all this information into a global world model. Using the explicit model of the environment built by these components, we have implemented path planning, simple ball handling skills and basic multi-agent cooperation. The resulting system is a very successful robotic soccer team, which has not lost any official game yet.

1 Introduction

Robotic soccer is a challenging research domain because problems in robotics, artificial intelligence, multi-agent systems and real-time reasoning have to be solved in order to create a successful team of robotic soccer players [12]. We took up the challenge of designing a robotic soccer team for two reasons. First of all, we intended to demonstrate the advantage of our perception methods based on laser range finders [8, 9, 10], which make *explicit world modelling* and *accurate and robust self-localization* possible. Secondly, we intended to address the problem of multirobot sensor integration in order to build a *global world model*. Of course, in order to demonstrate the usefulness of both concepts, we also had to implement basic ball handling skills, deliberation, and multi-agent cooperation that exploit the world model.

In a paper describing challenge tasks for robotic soccer, Asada *et al.* [2] conjectured that range finding devices are not sufficient for discriminating the ball, obstacles, and the goal [2, p.48]. Further, it was conjectured in this paper that a "conventional" approach of building an explicit world model, planning in this model, and executing the plan is not suitable for the dynamically changing environment in robotic soccer [2, p.49].

While we certainly agree that sonar sensors are not accurate and reliable enough, laser range-finders are definitely adequate for recognizing everything on the soccer field except the ball. Further, this can be easily used to construct an explicit world model which can support sophisticated behaviors and some form of explicit deliberation – provided deliberation is tightly coupled to observations.

As a matter of fact, we believe that building an explicit world model and using it for deliberation is a *necessary* prerequisite for playing an aesthetic and effective game of soccer. This conjecture is justified by the fact that the two winning teams in the simulation and the small size league in RoboCup'97 used this approach [5, 17]. The performance of these two teams were in sharp contrast to the teams in the middle size league at RoboCup'97. Although much of the unsatisfying performance in the middle size league could be probably attributed to problems concerning radio communication and problems due to the lightening conditions [16], some of it was probably also caused by the lack of an explicit world model. Further evidence for our claim is the performance of our team at RoboCup'98, which won the competition in the middle size league.

The rest of the paper is structured as follows. In the next section, we give an overview of the robot hardware and general architecture of our soccer team.

*This work has been partially supported by *Deutsche Forschungsgemeinschaft* (DFG) as part of the graduate school on *Human and Machine Intelligence*, by *Medien- und Filmgesellschaft Baden-Württemberg mbH* (MFG), and by *SICK AG*, who provided the laser range finders.

Section 3 focuses on our self-localization approach and Section 4 describes our player and ball recognition methods that are needed to create the local and the global world model. In Section 5 we describe the behavior-based control of the soccer agents and show how a basic form of multi-agent cooperation is achieved. Section 6 focuses on planning motion sequences that are needed to execute some of the behaviors. Finally, in Section 7 we describe our experience of participating in RoboCup'98 and in Section 8 we conclude.

2 Robot Hardware and General Architecture

Because our group is not specialized in developing robot platforms, we used an off-the-shelf robot—the *Pioneer 1* robot developed by Kurt Konolige and manufactured by *ActivMedia*.[1] In its basic version, however, the Pioneer 1 robot is hardly able to play soccer because of its limited sensory and effectory skills. For this reason, we had to add a number of hardware components (see Fig. 1).

Figure 1: Three of our five our robots: Two field players and the goal keeper

On each robot we mounted a *PLS200* laser rangefinders manufactured by *SICK AG* and a video camera connected to the *Cognachrome* vision system manufactured by *Newton Lab.*,[2] which is used to identify and track the ball. For local information processing, each robot is equipped with a *Toshiba* notebook *Libretto 70CT* running *Linux*. The robot is controlled using *Saphira* [13] that comes with the Pioneer robots. In order to enable communication between the robots and an off-field computer, we use the *WaveLan* radio ethernet. Finally, we added custom designed kickers and ball steering mechanisms to our robots.

The general architecture of each soccer agent (see Fig. 2) and the entire team (see Fig. 3) is very similar to those of other teams in the middle size league [1, 11]. However, there are also some noticeable differences.

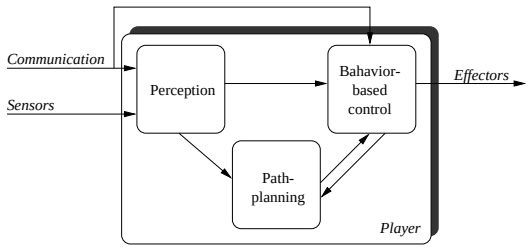

Figure 2: Player architecture

Our robots are basically autonomous robotic soccer players. They have all sensors, effectors and computers on-board. Each soccer agent has a *perception module* that builds a local world model (see Fig. 2). Based on the observed state of the world and intentions of other players communicated by the radio link, the *behavior-based control module* decides what behavior is activated. If the behavior involves moving to a particular target point on the field, the *path-planning* module is invoked which computes a collision-free path to the target point.

In order to initialize the soccer agents, to start and to stop the robots, and in order to monitor the state of all agents, we use a radio ethernet connection between the on-board computers and an off-field computer (see Fig. 3). If the radio connection is unusable, we still can operate the team by starting each agent manually. Most of the other teams in the middle size league use a very similar approach [1, 11].

Unlike other teams, we use the off-field computer and the radio connection for realizing *global sensor integration*, leading to a *global world model*. This world model is sent back to all players and they can employ this information to extend their own local view of the world. This means that the world model our players have is very similar to the world model constructed by an overhead camera as used in the small size league by teams such as *CMUnited* [17]. It should be noted, how-

[1] At this point we would like to thank *ActivMedia* for their timely support, resolving some of the problems which occured just a few weeks before RoboCup'98.

[2] *Newton Lab.* was also quite helpful in solving problems we had with their vision boards a few weeks before the tournament.

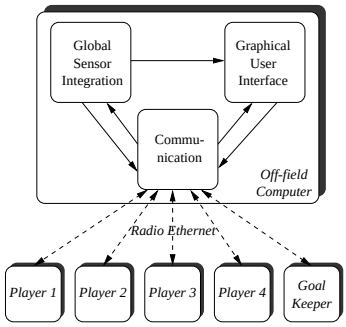

Figure 3: Team architecture

ever, that the information in our global world model is less accurate than the information obtained by direct observation (see Section 4).

3 Self-Localization

We started the development of our soccer team with the hypothesis that it is an obvious advantage if the robotic soccer agents know their position and orientation on the field. Based on our experience with different *self-localization* methods using laser range finders [8], we decided to employ such a method as one of the key components in our soccer agents.

There exist a number of different self-localization methods based on laser scans [4, 7, 10, 15, 19]. However, most of these methods are only *local*, i.e., they can only be used to correct an already existing position estimation. This means that once a robot looses its position, it will be completely lost. Further, all the methods are computationally very demanding, needing 100 msecs up to a few seconds. For these reasons we designed a new self-localization method which trades off generality for speed and the possibility of *global self-localization*.

Our method first extracts line segments from laser range scans and matches them against an *apriori* model of the soccer field. In order to ensure that extracted lines really correspond to field-border lines, only scan lines significantly longer than the extend of soccer robots are considered. The following algorithm shows how a set of position hypothesis is computed by recursively trying all pairings between scan lines and model lines:

Algorithm 1 *PositionHypothesis(M, S, Match)*
Input: model lines M, scan lines S, correspondence set $Match$
Output: set of positions hypothesis H

if $|Match| = |S|$ **then**
  $H := \{FitMatch(M, S, Match)\}$
else
  $H := \{\}$
  $s := SelectScanline(S, Match)$
  for all $m \in M$ **do**
    if $VerifyMatch(M, S, Match, m, s)$ **then**
      $H := H \cup$
      $PositionHypothesis(M, S,$
        $Match \cup \{m, s\})$
return H

The *FitMatch* function computes a position hypothesis from the *Match* set, *SelectScanline* selects the next scan line that should be matched, and *VerifyMatch* verifies that the pairings in *Match* are possible. This method is similar to the scan matching method described by Castellanos *et al.* [6]. In contrast to this approach, however, we only verify that the *global constraints* concerning translation and rotation as well as the length restrictions of scan lines are satisfied. This is sufficient for determining the position hypothesis and more efficient than Castellanos *et al.* approach.

Although it looks as if the worst-case runtime of the algorithm is $O(|M|^{|S|})$, it turns out that because of the geometric constraints the algorithm runs in $O(|M|^3|S|^2)$ time—provided the first two selected scan lines are not collinear or parallel [18]. For the RoboCup field the algorithm is capable of determining the global position of the robot modulo 180°—provided three field borders are visible.

For robust and accurate self-localization, the position information from odometry and scan matching is fused by using a Kalman filter. Therefore the probability that the robot is at a certain position l is modelled as a single Gaussian distribution:

$$l \sim N(\mu_l, \Sigma_l)$$

Here $\mu_l = (x, y, \alpha)^T$ is the mean value (the position with the highest probability) and Σ_l its 3×3 covariance matrix.

On robot motion $a \sim ((d, \theta)^T, \Sigma_a)$ where the robot moves forward a certain distance d and then rotates by θ, the position is updated according to:

$$\mu_l := E(F(l, a)) := \begin{pmatrix} x + d\ cos(\alpha) \\ y + d\ sin(\alpha) \\ \alpha + \theta \end{pmatrix}$$

$$\Sigma_l := \nabla F_l \Sigma_l \nabla F_l^T + \nabla F_a \Sigma_a \nabla F_a^T$$

Here E denotes the expected value of the function F and ∇F_l and ∇F_a are its Jacobians.

From scan matching a position estimate $s \sim N(\mu_s, \Sigma_s)$ is obtained and the robot position is updated using the formulas:

$$\begin{aligned} \mu_l &:= (\Sigma_l^{-1} + \Sigma_s^{-1})^{-1} \cdot (\Sigma_l^{-1}\mu_l + \Sigma_s^{-1}\mu_s) \\ \Sigma_l &:= (\Sigma_l^{-1} + \Sigma_s^{-1})^{-1} \end{aligned}$$

To initialize the self-localization system, a predefined value for μ_l is chosen and the diagonal elements of Σ_l are set to infinity. For the specific RoboCup environment, this ensures global self-localization on the first scan match.

The self-localization algorithm can then be implemented in a straightforward way. From a set of position hypotheses generated by the *PositionHypothesis* algorithm, the most plausible one is selected and Kalman-fused with the odometry position estimate. The robot position is then updated taking into account that the robot has moved since the scan was taken and matched.

Our hardware configuration allows 5–8 laser scans per second using only a few milliseconds for computing position hypotheses and the position update. Although a laser scan may include readings from objects blocking the sight to the field borders, we didn't experience any failures in the position estimation process. In particular, we never observed the situation that one of our robots got its orientation wrong and "changed sides."

4 Building Local and Global World Models

Each soccer agent interprets its sensor inputs using the perception module shown in Fig. 4 in order to estimate its own position, the position of observed players and the ball position.

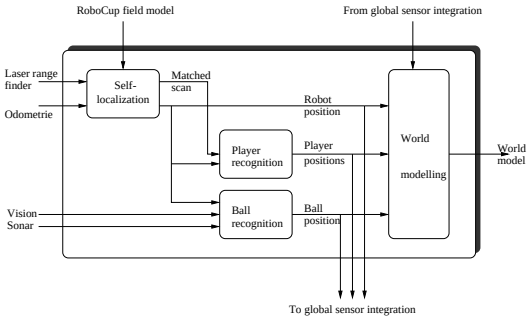

Figure 4: Perception module

After the self-localization module matched a range scan, scan points that correspond to field lines are removed and the remaining points are clustered. For each cluster the center of gravity is computed and interpreted as the approximate position of a robot. Inherent to this approach is a systematic error depending on the shape of the robots.

Since our laser range finders are mounted well above the height of the ball, we cannot use it for ball recognition. In fact, even if it were mounted lower, it is questionable whether it would be possible to distinguish the ball from the players by shape. For this reason, we use a commercially available vision system for ball recognition.

If the camera sees an object of a trained color (a so-called *blob*), the vision system outputs the pixel coordinates of the center of the blob, its width, height and area size. From these pixel coordinates we compute the relative position of the ball with respect to the robot position by mapping pixel coordinates to distance and angle. This mapping is learned by training the correspondence between pixel coordinates and angles and distances for a set of well-chosen real-world positions and using interpolation for other pixels. In order to improve the quality of the position estimation, we use the sonar sensors as a secondary source of information for determining the ball position.

From the estimated position of the player, the estimated position of other objects and the estimated position of the ball – if it is visible – the soccer agent constructs its own *local world model*. By keeping a history list of positions for all objects, their headings and velocities can be determined. To reduce noise, headings and velocities are low-pass filtered. Position, heading, and velocity estimates are sent to the *multi-robot sensor integration module*.

In addition to objects that are directly observable, the local world model also contains information about objects that are not visible. First of all, if an object disappears temporarily from the robot's view, it is not immediately removed from the world model. Using its last known position and estimated heading and velocity, its most likely position is estimated for a few seconds. Secondly, information from the global world model is used to extend the local world model of a player. Objects of the global world model which don't correspond to any object of the local world model are added to the local world model, but marked as not really visible for the player. If an object of the global world model corresponds to an object of the local model the local information regarding exact position, heading end velocity is given priority because it is probably more recent and accurate. In this case the global information is only used to determine the

objects identity.

The *global world model* is constructed from time-stamped position, heading, and velocity estimates that each soccer agent sends to the global sensor-integration module. Using these estimates, it is easy to tell whether an observed object is friend or foe. Knowing who and where the team members are is, of course, very helpful in playing a cooperative game.

A further information that is very useful is the global ball position. Our vision hardware recognizes the ball only up to a distance of 3–4 m. Knowing the global ball position even if it is not directly visible enables the soccer robot to turn its camera into the direction of where the ball is expected avoiding a search for the ball by turning around. This is important in particular for the goal keeper, which might miss a ball from the left while it searches for the ball on the right side.

It should be noted, however, that due to the inherent delay between sensing an object and receiving back a message from the global sensor integration, the information from the global world model is always 100–400 msecs old. This means that it cannot be used to control the robot behavior directly. However, apart from the two uses spelled out above, there are nevertheless a number of important problems that could be solved using this global world model – and we will work on these points in the future. Firstly, the global world model could be used to reorient disoriented team members. Although we never experienced such a disorientation, such a fall-back mechanism is certainly worthwhile. Secondly, it provides a way to detect unreliable sensor systems of some of the soccer agents. Thirdly, the global world model could be used for making strategic decisions, such as changing roles dynamically [17].

5 Behavior-based Control and Multi-Agent Cooperation

The soccer agent's decisions are mainly based on the situation represented in the explicit world model. However, in order to create cooperative team behavior, actual decision are also based on the *role* assigned to the particular agent and on intentions communicated by other players.

Although the control of the execution can be described as behavior-based, our approach differs significantly from approaches where behaviors are activated by uninterpreted sensor inputs [3]. In our case, high-level features that are derived from sensor inputs and from the communication with other agents determine what behavior is activated. Furthermore, behaviors may invoke significant *deliberation* such as planning the path to a particular target point (see Section 6).

5.1 Basic Skills and Behavior-Based Control

The behavior-based control module consists of a rule-based system that maps situations to actions. The rules are evaluated every 100 msecs so that the module can react immediately to changes in the world. Depending on whether the agent fills the *role* of the goal keeper or of a field player, there are different rule sets.

The goalie is very simple minded and just tries to keep the ball from rolling into our goal. It always watches the ball – getting its information from the global world model if the camera cannot recognize the ball – and moves to the point where the robot expects to intercept the ball based on its heading. If the ball is on the left or right of the goal, the goal keeper turns to face the ball. In order to allow for fast movements, we use a special hardware setup where the "head" of the goalie is mounted to the right as shown in Fig. 1. If the ball hits the goalie, the kicking device kicks it back into the field.

The field players have a much more elaborate set of skills. The first four skills below concern situations when the ball cannot be played directly, while the two last skills address ball handling:

Approach-position: Approach a target position carefully.

Go-to-position: Plan and constantly re-plan a collision-free path from the robot's current position to a target position and follow this path until the target position is reached.

Observe-ball: Set the robots heading such that the ball is in the center of focus. Track the ball without approaching it.

Search-ball: Turn the robot in order to find the ball. If the ball is not found after one revolution go to *home position* and search again from there.

Move-ball: Determine a straight line to the goal which has the largest distance to any object on the field. Follow this line at increasing velocity and redetermine line whenever appropriate.

Shoot-ball: To accelerate the ball either turn the robot rapidly with the ball between the flippers or

use the kicker-mechanism. The decision on which mechanism to use and in which direction to turn is made according to the current game situation.

The mapping from situations to actions is implemented in a decision-tree like manner as shown in Fig. 5. Taking into account the currently executed

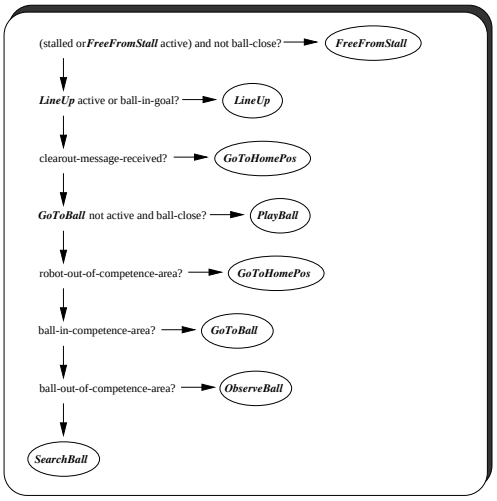

Figure 5: Decision tree for action selection. Left arrow: yes, down arrow: no, circle: new state

action and the current world model the rules are permanently evaluated leading to a decision which action to take next. Possible actions include:

FreeFromStall: Select and follow a path to a clear position on the field.

GoToHomePos: Go to the home position using the *go-to-position* skill.

LineUp: "Be happy"(play music and turn on the spot), then go to the home position.

SearchBall: Invoke the *search-ball* behavior.

PlayBall: Attack using the *approach-position*, *move-ball* and *shoot-ball* skills.

ObserveBall: Track the ball using the *observe-ball* behavior.

GoToBall: Go to the ball using the *go-to-position* and *approach-position* behaviors.

It should be noted that details of tactical decisions and behaviors were subject to permanent modifications even during RoboCuop'98. As a reaction to teams which would just push the ball and opponents over the field we modified our stall behavior to not yield in such situations. Unfortunately the capability to recognize when a goal was shot and to line up to wait for game start was of no use since the field got too crowded with people repositioning their robots after our team scored a goal.

5.2 Multi-Agent Coordination

If all of the soccer player would act according to the same set of rules, a "swarm behavior" would result, where the soccer players would block each other. One way to solve this problem is to assign different *roles* to the players and to define *areas of competence* for these roles (see Fig. 6). If these areas would be

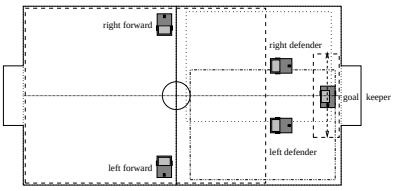

Figure 6: Roles and areas of competence

non-overlapping, interference between team members should not happen, even without any communication between players. Each player would go to ball and pass it on to the next area of competence or into the goal. In fact, this was our initial design and it is still the fall-back strategy when the radio communication is not working.

There are numerous problems with such a rigid assignment of competence areas, however. Firstly, players may interfere at the border lines between competence areas. Secondly, if a player is blocked by the other team, broken, or removed from the field, no player will handle balls in the corresponding area. Thirdly, if a defender has the chance of dribbling the ball to the opponent's goal, the corresponding forward will most probably block this run. For these reasons, we modified our initial design significantly.

If a player is in a good position to play the ball it sends a **clear-out** message. As a reaction to receiving such a message other players try to keep out of the playing robots way (see Fig. 5). This helps to avoid situations in which two team mates block each other. Based on communicating intentions, areas of competence can be made overlapping as shown in Fig. 6. Now, the forwards handle three quarters of the field and attacks are coordinated by exchanging the intentions.

We do not have any special coordination for defensive moves. In fact, defensive behavior *emerges* from

the behavior-based control described above. When the ball enters our half of the field, our defenders go to the ball and by that block the attack. Surprisingly, this simple defensive strategy worked quite successfully.

6 Path Planning

Some of the skills described in the last section concern the movement of the soccer robots to some target point on the field. While such movements could be realized in a behavior-based way, we decided to *plan* the motion sequence in order to avoid problems such as local minima.

Motion planning in the presence of moving obstacles is known to be a computationally very demanding problem [14]. Furthermore, because the movements of the opponents are hardly predictable, a motion plan would be probably obsolete long before it has been generated. For these reasons, we decided to approximate the solution to the motion planning problem with moving obstacles by solving the path planning problem with stationary obstacles. Although such an approach might seem to be inadequate in an environment that is as dynamic as robotic soccer, experience shows that often enough the opponent players can indeed be approximated as stationary obstacles. More importantly, however, our path planning method is so efficient – needing only a few milliseconds for 4–5 obstacles – that constant re-planning is possible.

To plan arbitrary paths around objects, we use the *extended visibility graph method* [14]. Objects in the world model are grown and the soccer field is shrunken allowing path planning for robot shrunken to point. The actual planning is done by an A^* algorithm that finds shortest collision-free paths consisting of straight line and arc segments from the current robot position to the desired target position.

To increase speed in planning, an iterative planing approach is used. Beginning with only the start and goal node, objects are only added to the graph if they interfere with a found path. To avoid oscillation between paths with similar costs, a distance penalty is added to paths which require large changes of the robots heading in the beginning.

7 Experience at RoboCup'98

Participating in the RoboCup'98 tournament was very fruitful for us in two ways. Firstly, we got the opportunity to exchange ideas with other teams and learned how they approached the problems. Secondly, we learned much from playing. As pointed out at various places in the paper, we redesigned tactics and strategy during the tournament incorporating the experience we made during the games.

Our experience with hard- and software reliability was mixed. The laser range finders and our self-localization worked without any problem, while the radio communication was sometimes jammed, perhaps by other teams playing at the same time on another field. The most fragile part was our vision system – not because of hardware failures, but because slightly changed ligthening conditions led to serious problems in ball recognition. However, this seemed to be a problem for all teams.

The performance of our team at RoboCup'98 was quite satisfying. Apart from winning the tournament, we also had the best goal difference (12:1), never lost a game, and scored almost 25% of the goals during the tournament. That this performance was not accidental was demonstrated at the *German Open VISION-RoboCup'98*, October 1998, in Stuttgart. Again, we won the tournament and did not loose any game.

The key components for this success are most probably the *self-localization* and *object-recognition* techniques based on laser range finders, which enable us to create accurate and reliable *local* and *global world models*. Based on these world models, we were able to implement accurate, reactive path-planning, fine-tuned behaviors, and basic multi-agent cooperation – which was instrumental in winning. Finally, the mechnical design of our kicker and the ball steering mechanism certainly also played a role in playing successful robotic soccer.

8 Conclusions and Future Directions

Robotic soccer is a challenging research domain. In this context, we addressed the problem of building an accurate and reliable world model for each soccer agent using laser range finders and to integrate these into a *global world model*. Based on these explicit world models, simple soccer skills, accurate path planning, and multi-agent cooperation was realized. The resulting system is a very successful robotic soccer team, which has not been beaten yet in an official game. There are nevertheless a number of points where significant improvements are possible. For instance, we plan to improve

- the low-level sensor interpretation by exploiting more features of our hardware and by using real-

time extensions of the Linux system for getting precise time-stamps of sensor measurements;

- the accuracy and robustness of multirobot sensor integration;
- the low-level control of the movements in order to get smoother behaviors;
- the soccer skills based on the above improvements, e.g., to realize ball passing;
- the strategic decision making by allowing for flexible role assignments and by using explicit deliberation based on the current global state.

Summarizing, we hope that we will be able to demonstrate that our robots are able to play even more effective and aesthetic robotic soccer at RoboCup'99.

References

[1] M. Asada, editor. *RoboCup-97: Robot Soccer World Cup II*. Springer-Verlag, New York, 1998, to appear.

[2] M. Asada, P. Stone, H. Kitano, A. Drogoul, D. Duhaut, M. Veloso, H. Asama, and S. Suzuki. The RoboCup physical agent challenge: Goals and protocols for phase I. In Kitano [11], pages 42–61.

[3] R. A. Brooks. A robust layered control system for a mobile robot. *IEEE Journal of Robotics & Automation*, RA-2(1), 1986.

[4] W. Burgard, D. Fox, D. Hennig, and T. Schmidt. Estimating the absolute position of a mobile robot using position probability grids. In *Proc. AAAI-96*, p. 896–901, Portland, OR, July 1996. MIT Press.

[5] H.-D. Burkhard, M. Hannebauer, and J. Wendler. AT Humbold – development, practice and theory. In Kitano [11], p. 357–372.

[6] J. A. Castellanos, J. D. Tardós, and J. Neira. Constraint-based mobile robot localization. In *International Workshop on Advanced Robotics and Intelligent Machines*. Univ. of Salford, Apr. 1996.

[7] I. J. Cox. Blanche: Position estimation for an autonomous robot vehicle. In I. J. Cox and G. T. Wilfong, editors, *Autonomous Robot Vehicles*, p. 221–228. Springer-Verlag, New York, 1990.

[8] J.-S. Gutmann, W. Burgard, D. Fox, and K. Konolige. An experimental comparison of localization methods. In *Proc. IROS'98*. IEEE/RSJ, 1998.

[9] J.-S. Gutmann and B. Nebel. Navigation mobiler Roboter mit Laserscans. In P. Levi, T. Bräunl, and N. Oswald, editors, *Autonome Mobile System 1997*, p. 36–47, 1997. Springer-Verlag.

[10] J.-S. Gutmann and C. Schlegel. Amos: Comparison of scan matching approaches for self-localization in indoor environments. In *Proc. 1st Euromicro Workshop on Advanced Mobile Robots*, pages 61–67. IEEE, 1996.

[11] H. Kitano, editor. *RoboCup-97: Robot Soccer World Cup I*, Springer-Verlag, New York, 1998.

[12] H. Kitano, M. Asada, Y. Kuniyoshi, I. Noda, E. Osawa, and H. Matsubara. RoboCup: A challenge problem for AI. *The AI Magazine*, 18(1):73–85, 1997.

[13] K. Konolige, K. Myers, E. H. Ruspini, and A. Saffiotti. The Saphira Architecture: A Design for Autonomy. *Journal of Experimental and Theoretical Artificial Intelligence*, 9:215–235, 1997.

[14] J.-C. Latombe. *Robot Motion Planning*. Kluwer, Dordrecht, Holland, 1991.

[15] F. Lu and E. E. Milios. Robot pose estimation in unknown environments by matching 2D range scans. In *Proc. CVPR-94*, pages 935–938, 1994.

[16] I. Noda, S. Suzuki, H. Matsubara, M. Asada, and H. Kitano. Overview of RoboCup-97. In Kitano [11], pages 20–41.

[17] M. Veloso, P. Stone, K. Han, and S. Achim. The CMUnited-97 small robot team. In Kitano [11], pages 242–256.

[18] T. Weigel. Roboter-Fußball: Selbstlokalisation, Pfadplanung und Basisfähigkeiten. Diplomarbeit, Fakultät für Angewandte Wissenschaften, Freiburg, Germany, 1998. In preparation.

[19] G. Weiß and E. von Puttkamer. A map based on laserscans without geometric interpretation. In U. Rembold, R. Dillmann, L. Hertzberger, and T. Kanade, editors, *Proc. IAS-95*, pages 403–407. IOS Press, 1995.