Football is coming Home

Stefan Schiffer, Alexander Ferrein, and Gerhard Lakemeyer

Knowledge-Based Systems Group
Computer Science Department
RWTH Aachen University
Aachen, Germany
{schiffer,ferrein,gerhard}@cs.rwth-aachen.de

Abstract. Most of the robots in the RoboCup soccer league are made especially for the task of playing soccer. They use methods that are specifically designed for the soccer domain and would perhaps fail in other robotic testbeds such as the newly established RoboCup@Home league without making fundamental changes throughout their entire software system. In contrast, our robots and the control software were designed with a broader field of application in mind. This paper sketches our way from the soccer application to the RoboCup@Home league.

1 Introduction

Research on mobile robots has been done for over 30 years now. A decade ago researchers started to establish a common test bed for research on mobile robotics and agent systems: RoboCup. The application domain chosen was soccer which provides many interesting aspects: it is an adversarial multi-agent real-time domain. To reach a broader scope, beside different soccer leagues a Rescue league was installed. Here, mobile robots have to save entombed people from urban disaster areas. Another new league, called RoboCup@Home, was established this year which focuses on service robotics in home-like environments.

One interesting question is how results from soccer can be transfered to other fields and applications. At the RoboCup championships this year we participated in both, the Middle-size soccer league, and in the new service robotics league RoboCup@Home with the same robots. In this paper we report on our approach to the soccer problem and report on how we adapted the soccer robot to be able to participate also in the RoboCup@Home league. We use ReadyLog [1, 2], a derivative of the logical robot programming language Golog [3], to implement the behavior of the soccer robots. To be able to specify the behavior of a soccer robot in a very natural, human-like fashion, we developed a qualitative world model that supports the formulation of soccer moves. Very much of the development made for playing soccer could be directly adopted to the service robots. Some new developments had to be made, though. Of special interest in the service robotic league is the adaptability of the robot system to the environment as well as human machine interaction. We developed a map building tool which allows to easily build up the environment map with semantic annotations like

"fridge" or "living room". To settle tasks to the robot we integrated existing speech recognition and speech synthesis systems to provide a natural human machine interface.

The paper is organized as follows. In Section 2 we briefly describe our robot system. Section 3 shortly describes the language ReadyLog and shows how behaviors can be specified. Further, we present the qualitative world model abstraction and show how it can ease the specification. In Section 4 we introduce the RoboCup@Home league in greater detail and discuss the map building tool and show the human machine interface. We conclude in Section 5.

2 Mobile Robotics

In this section we give an overview of our robot and the low-level control software. In particular, we address the topics of localization, navigation, and computer vision, which are fundamental problems that have to be solved for a mobile robot.

2.1 Hardware

Our hardware platform has a size of 40 cm × 40 cm × 60 cm. It is driven by a differential drive, the motors have a total power of 2.4 kW and are originally developed for electric wheel chairs. For power supply we have two 12 V lead-gel accumulators with 15 Ah each on-board. The battery power lasts for approximately one hour at full charge. This power provides us with a top speed of 3 m/s and 1000°/s at a total weight of approximately 60 kg. On-board we have two Pentium III PC's at 933 MHz running Linux, one equipped with a frame-grabber for a Sony EVI-D100P camera mounted on a pan/tilt unit and an omni-vision camera. Our other sensor is a 360° laser range finder (LRF) with a resolution of 1 degree at a frequency of 10 Hz. For communication a WLAN adapter based on IEEE 802.11b is installed. This platform can be utilized for playing soccer but it is also possible to use it for service robotics applications.

2.2 Localization

As described in [4] we make use of the Monte Carlo Localization (MCL) algorithm [5] for our self-localization. It works by approximating the position estimation by a set of weighted samples: $\mathbf{P}(l_t) \sim \{(l_{1,t}, w_{1,t}), \ldots, (l_{N,t}, w_{N,t})\} = \mathbf{S}_t$. Each sample represents one hypothesis for the pose of the robot. Roughly, the Monte Carlo Localization algorithm chooses the most likely hypothesis given the previous estimate, the actual sensor input, the current motor commands, and a map of the environment. Imagine, for example, a global localization on a soccer field. In the beginning the robot has no clue about its position and therefore it has many hypotheses. After driving around and taking new sensor updates the robot's belief about its position condenses to two main hypotheses. This is

due to the symmetry of the soccer field. The robot cannot resolve this ambiguity with the laser scanner. However, it is possible to resolve it with additional information provided by the camera, for example.

The method is presented in detail in [4]. It was developed for the RoboCup soccer setting in the first place, but it works very well and yet better for indoor navigation even in large environments. That is because maps in those domains contain a larger number of structural features that can be used for localization.

2.3 Collision Avoidance and Navigation

Our collision avoidance module performs an A* search over an occupancy grid [6] which is generated from the laser scanner inputs. The robot is positioned in the middle (origin) of such a grid. A collision-free path to the desired destination is calculated by performing an A* search from the robot's current location to the given target point. If the target point is located outside the grid range, we project the target point onto the border of the grid. To alleviate the search we extend the cells occupied by an obstacle by the size of our robot. Thus, the robot can be regarded as a mass point. The possible actions for the search are $A = \{N, S, W, E, NW, SW, ...\}$, i.e. the robot can move to any neighboring cell. To apply A* we need to provide a cost function and a heuristic function. The cost function is the Euclidean distance between grid cells. As heuristic function we use the Manhattan distance to the target point.

The path calculated by A* must be translated into motor commands. Thus, we need a curve from which we can derive the appropriate commands sent to the motors. We approximate the steering commands by again applying an A* search, this time over the acceleration space. From the results appropriate translational and rotational velocities are found allowing the robot to safely drive to the given target point. Since some objects cannot be detected with the laser range finder we employ a so-called obstacle server that uses an additional map of the environment. This map contains areas that the robot is not allowed to enter but which cannot be seen with the LRF. These areas are integrated as obstacles into the robots local perception. Thus, it can avoid the regions although it is not able to 'see' them.

2.4 Vision

In soccer, the ball is detected by our *vision* module. The camera image is inspected on several scan lines. If a sufficient number of pixels on a scan line has the appropriate object color we grow a region of interest around these pixels. Each region is then color segmented. The segmentation is based on a color map which is gained in a color calibration process, where the different colors are trained in a supervision mode on a few sample images. The thresholds for the different colors are found following a Bayesian approach. For finding the ball we apply a randomized circle fitting following [7]. The circle fitting is implemented as an any-time algorithm which returns the best fitted circle. With a geometric model of the robot the position of the ball is estimated.

3 Robotic Soccer

Is robotic soccer a domain for cognitive robots? Isn't it sufficient to implement a reactive or rule-based system for the simple actions like "dribble" or "kick" which are taken in the soccer domain? Raising these questions is allowed, and yes, one does not need formal knowledge representation and reasoning techniques to attack the problem of soccer playing agents or robots. But these kind of techniques can be applied, and they even pay off. During the recent years we participated in RoboCup's Middle-Size league with our soccer robots following such an approach. We used a derivative of the logic-based language GOLOG for programming our robots. Further, it has been shown in [8] that GOLOG is also suitable for specifying tactics and strategies in soccer which then can be transfered to the robot.

In the following we briefly introduce READYLOG, our variant of GOLOG which we use for behavior specification on our soccer robots. Then we present a formal approach to the specification of soccer moves and strategies as described in [8]. To make these formalizations applicable for soccer robots, we show how the quantitative data gathered from the sensor system of the robot can be abstracted to a qualitative world model which supports the task of formulating soccer moves.

3.1 Behavior Specification with ReadyLog

READYLOG [1, 2], a variant of GOLOG, is based on Reiter's variant of the Situation Calculus [3, 9], a second-order language for reasoning about actions and their effects. Changes in the world are only due to actions so that a situation is completely described by the history of actions starting in some initial situation. Properties of the world are described by *fluents*, which are situation-dependent predicates and functions. For each fluent the user defines a successor state axiom specifying precisely which value the fluent takes on after performing an action. These, together with precondition axioms for each action, axioms for the initial situation, foundational and unique names axioms, form a so-called *basic action theory* [9].

GOLOG emerged to an expressive language over the recent years. It has imperative control constructs such as loops, conditionals [10], and recursive procedures, but also less standard constructs like the nondeterministic choice of actions. Extensions exist for dealing with continuous change [11] and concurrency [12], allowing for exogenous and sensing actions [13] and probabilistic projections into the future [14], or decision-theoretic planning [15] which employs Markov Decision Processes (MDPs).

READYLOG integrates these extensions in one agent programming framework [1, 2]. For specifying the behaviors of an agent or robot the following constructs exist: (1) sequence $(a; b)$, (2) nondeterministic choice between actions $(a|b)$, (3) solve an MDP ($solve(p, h)$, p is a GOLOG program, h is the MDP's solution horizon), (4) test actions ($?(c)$), (5) event-interrupt ($waitFor(c)$), (6) conditionals ($if(c, a_1, a_2)$), (7) loops ($while(c, a_1)$), (8) condition-bounded execution ($withCtrl(c, a_1)$), (9) concurrent execution of programs ($pconc(p_1, p_2)$),

(10) probabilistic actions ($prob(val_{prob}, a_1, a_2)$), (11) probabilistic (offline) projection ($pproj(c, a_1)$), and (12) procedures ($proc(name(parameters), body)$).

To encode the behavior one has to give a domain axiomatization including the actions the robot can perform together with their effects, and the fluents which describe the properties of the world like the ball position. Examples of domain descriptions for the soccer domain can be found in [16, 17].

3.2 Qualitative World Model Representation

In [8], Dylla et al. use GOLOG as the specification language for soccer moves. They looked at soccer literature to derive the tactics known from human soccer play and transfer it to soccer playing robots. When trying to perform this transfer a gap in how knowledge is represented in human notions and with robot systems becomes obvious. The notions for soccer moves Dylla et al. base on have a qualitative nature. For the robots we need a numerical representation instead. To close this gap we started to develop a qualitative world model for soccer robots. From the quantitative sensor data the robot gathers we calculate a set of qualitative predicates. Here, we will only give a brief overview of some of these predicates. A concise description can be found in [18]. The selection of predicates is tailored for formulating soccer moves. Clearly, the use of qualitative predicates for robots and agent systems, in general, is quite common.

The basic notion needed for describing soccer moves is a notion of space and distance. The approach we describe in [18] is based on [19, 20]. We started with an egocentric relation of distance. One defines a metric for $\mathbb{R}^1$, builds equivalence classes for ranges of $\mathbb{R}^1$ and assigns constants like "near" or "far" to each class. To build the orientation relation we build equivalence classes over ranges of angles. Each sector is assigned an orientation like with a compass rose. We use eight different orientations like "front", "front-right", "right" and so on. With these models for distance and orientation the robot is able to describe objects in a qualitative egocentric fashion like "the ball is located in the front-left direction at a medium distance". What is further needed to describe the soccer scenario is a qualitative notion of strategic positions in a global frame of reference. In soccer, there are several strategic roles a player can have like "defense", "midfield", and "offense". The area each member of one of these groups is located in is a strategic region on the field. Further, one can distinguish between three sides of the pitch. The play can be on the left or right side, or in the center of the field. For our qualitative world model we defined five zones ranging from $farFront$, which is a zone direct to the opponent goal, to $farBack$ which includes the own goal, and the sides "left", "middle", "right". Again, we refer to [18] for details.

Having a description of positions of the field, one must also regard other tactical properties of soccer. As [8] worked out the notion of reachability is central for performing cooperative actions. As a mathematical model for reachability we use Voronoi Diagrams and their dual, the Delaunay triangulation.[1] For space reasons we omit other relevant qualitative predicates pointing to [18] for details.

[1] A Voronoi diagram $V(S)$ of a set S of n point sites is the partitioning of a plane with n points into n convex polygons such that each polygon contains exactly one

```
1   proc build_up_play_bestInterceptor
      if haveBall(ownNumber) then
        getFreeSide(offense, freeSector); getPassPartner(offense, freeSector, passPartner);
        solve( 3,
          if ¬isKickable(ownNumber) then intercept(ownNumber)
6         else if isKickable(ownNumber)∧ isPassReachable(ownNumber, passPartner) then
                passTo(ownNumber, passPartner)
            endif
          else
          | dribbleTo(ownNumber, freeSector)
11        | pickBest(bestSide, {leftSide, middleSide, rightSide}
            dribbleTo(ownNumber, bestSide, middleZone)
            | kickTo(ownNumber,freeSector) ) /*end pickBest*/
          | move_kick(bestSide, freeSector)
        else
16        intercept (ownNumber) | kickTo(ownNumber,freeSector)
        endif
        ) /*end solve with horizon 3*/
      else
        intercept (ownNumber)
21    endif /*end if haveBall*/
    endproc /*end proc build_up_play*/
```

Fig. 1. Exemplary ReadyLog program for qualitative build up play

To give an impression how the qualitative world model eases the specification of the behavior of the robot in Fig. 1 we show an exemplary ReadyLog program where the qualitative world model is used. The situation is that one player is to make a goal kick from the own goal. If the ball is not kickable, the robot first intercepts the ball. Then it checks if one of the teammates is reachable with a pass, or if it should better take the ball and dribble towards a free region on the field. In the *pickBest* statement ReadyLog optimizes over the argument *bestSide* for the following actions. The arguments are chosen from the domain $\{leftSide, middleSide, rightSide\}$. As one can also see by the helper predicate *getPassPartner* in line 3 one does not have to specify particular coordinates but can use the qualitative predicates instead. This eases the specification in a way that the designer can formulate strategies in an abstract way. Introducing a world model abstraction layer also leads to more general code and it is easier to adapt the code to new environmental conditions like an increasing field size. Further, it is also advantageous for our decision-theoretic planning as it reduces the state space. Clearly, during execution the generated policy has to be refined and the abstracted positions have to be instantiated with the respective coordinates of the addressed objects.

Using qualitative predicates with ReadyLog one must be able to reason with them. Several calculi like the Region Connection Calculus (RCC) [22] are known but it was shown that they are computationally demanding (e.g. in [23] the RCC is proven being NP-complete). Using RCC only for querying predicates would be okay. Note that during projections we need more than database queries. As time constraints for generating the next action to take are strict in robotic

---

point and every point in the given polygon is closer to its central point than any other. For a more detailed account on Voronoi diagrams and their dual, the Delaunay triangulation $DT(S)$, see e.g. [21].

soccer we refrained to use one of these formal calculi. Instead we make use of the fact that we have both, a quantitative and a qualitative representation of our world. For each qualitative predicate we define an inverse function which maps the range of quantitative data abstracted by a qualitative predicate to one single value. For example, a tactical region (zone or side) is projected to the center of gravity of the region which is used as a representative for the respective region. Together with the projection mechanism in ReadyLog we are able to reason with the qualitative predicates (see [18] for a detailed example).

4 RoboCup@Home

RoboCup@Home is a new league in RoboCup which focuses on real world applications of robotics in home-like environments. Its aim is to foster the development of robotic applications that are useful and assisting in everyday life. Our robot platform was not exclusively designed for the soccer domain in the first place. Instead, it can also be used for service robotic tasks as well as for teaching purposes. It was of course not possible to participate right away, but the modifications needed to get started were of only smaller complexity. The requirements in the @Home league differ from those in the soccer domain. The necessary enhancements of our control software are directly conditioned by the characteristics of the RoboCup@Home league. To get a rough impression, we now briefly describe the basic setup and we point out the characteristics.

4.1 Characteristics

The @Home league chose 'the real world' as its application domain. To approach this goal the scenario features a basic home-like environment which will be gradually extended over the years. In general, no special preparations are made to or in the environment to make it more suitable for a robot. In particular, this means there are no markers or active landmarks nor is any of the objects in the environment color coded in any way. Furthermore, the lighting conditions are not standardized. The first RoboCup@Home competitions in Bremen were conducted in an area of about 50 m^2. The area was surrounded by walls including two doors as entrances at both ends. A room divider was used to form two separate areas one of which served as a kitchen and the other one being the dining room and the living room. All rooms were equipped with appropriate furniture such as tables, a couch, shelfs, lamps, and plants. The environment in Bremen is depicted in Figure 3(b).

The competition itself is organized in form of several specific tests, an open challenge, and a final round. The specific tests are meant to reveal a general level of performance of each participating robot. In the first year the tests included a follow challenge, a navigate challenge, and a manipulate challenge. Each of these challenges consists of two phases where the first phase represents a proof of concept. Teams are allowed to make some arrangements helping their robot to pass the test. The second phase should demonstrate the general applicability of the

approach and no modifications to the environment are allowed. In the open challenge as well as in the finals teams can freely choose what they want to show to the audience. The performance is evaluated by a jury which reviews the presentation based on criteria such as robustness, usefulness, difficulty, entertainment value, and alike.

The characteristics identifiable above demand for versatile and flexible approaches. Our basic framework already allows for robust navigation and collision avoidance within almost all indoor environments. Thus, me managed to do the navigate test with only small effort. We solved the follow challenge by using additional output from our localization module. It recognizes and classifies objects that do not belong to the map. We calibrate on the object we need to follow and track it over time using the Hungarian method [24]. Since our only actuator so far is our kicking device which is not very useful in a home environment we did not take the manipulate test. In the free challenges we combined the laser-based object detection and a slightly modified version of our vision to detect and retrieve one of two boxes of beverages previously hidden within the environment.

Interaction between the robot and humans is an integral part of each test in the RoboCup@Home competitions. One important thing when acting in a human populated environment is that the robot needs to 'know' about the environment in form of human terminology and concepts. To be able to do that the robot needs to understand which objects, places, or actions the human refers to e.g. in natural speech. To determine its own position and the position of other objects the robot relies on a map of the area it is supposed to act in. This map is generated or recorded in advance and this process can take quite some time. Moving to a scenario that should reflect a normal human living space as described above changes can occur frequently. Thus, time for the process of generating such a map is very limited. Still, a table being moved from one side of a room to the other still is the same table and should be recognized as such.

What we needed to do was to enrich the low-level data with meaningful semantic information such as common names in human terminology. Moreover, we needed a mechanism to quickly adapt to changes possibly made to the environment. In the following we present our solutions.

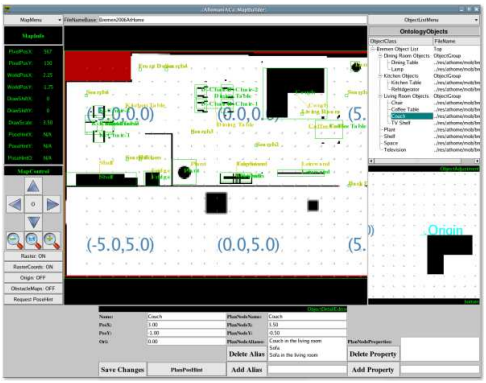
4.2 Map Building Tool

In order to be able to efficiently adapt to the frequent changes which are immanent in a home-like environment we developed a *semantic* map building application. It allows us to update the robot's world representation to the current situation very quickly. Our map builder uses a collection of semantically annotated objects that can be dragged and dropped to their specific location in a base-map. This simplifies the map building process to some few clicks. The objects and the ontology they are categorized in are managed in XML-based files. That is, there is a data type definition (DTD) which defines all relevant properties that are associated with an object. A sample definition is depicted in Figure 2. In our case it includes a signature of the object as seen by the laser range finder, the area to be used in the obstacle server, and a name along with

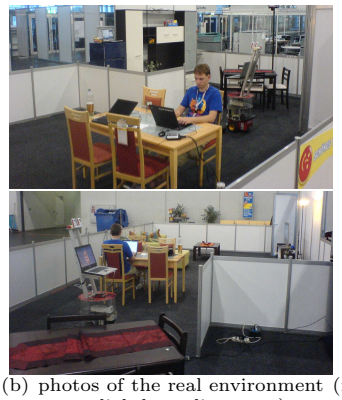
```
<!ELEMENT MapObject ( Name,
                      MapFileName,
                      ObsFileName,
                      Position,
                      PlanNodeName,
                      PlanNodeAlias* ) >
```

```
<!ELEMENT SEMap     ( Name,
                      Location,
                      BaseMapFileName,
                      Room*,
                      MapObject* ) >
```

Fig. 2. Excerpts from the DTDs of a map object and an overall map file.



(a) map building tool



(b) photos of the real environment (in a slightly earlier stage)

Fig. 3. A screenshot of our map building tool. It shows the creation of the map of the home environment at RoboCup 2006 in Bremen.

some common aliases. If additionally a vision system is used one could also include sample pictures of the respective object. The particular information for each object have to be provided beforehand, e.g. the signature of an object as seen by the laser range finder has to be drawn or recorded and pictures need to be taken and associated with the object.

Figure 3(a) shows a screenshot of the tool we developed for the map building task. Objects present in the current configuration of the environment can be dragged and dropped to their respective location. When the environment has been set up completely several low-level data files are generated, each with all the information required. That is, for localization with a laser range finder, for example, the signature of every object is integrated into the occupancy grid used for localization at the corresponding position. If necessary, a corresponding area is added to the obstacle map. Additionally, a node is added to a topological map which is needed for path planning. The items in the different low-level data structures are inter-referenced by their name. This way, each module can refer to an object or place by its name in human terminology.

By providing the robot with semantically enriched objects it is able to make use of any particular part of the information associated with an object. Thus, interaction with humans in the environment can be achieved in a transparent fashion. For instance, when a human specifies a target location for the robot to reach, the name recognized by the speech recognition module is passed to the path planning module which in turn instructs the navigation module to drive to the associated coordinate within the map. We now briefly describe how we realize the human robot interaction.

4.3 Human Machine Interaction

As already mentioned, in a natural human environment interaction between the robot and the human beings around it is an integral part of the challenges in the RoboCup@Home league. Therefore, we realized communication facilities in terms of a speech recognition module to process human instructions, requests, and questions. To inform, answer, or ask the human for clarification on the current task we also provide a speech synthesis module which enables the robot to attract attention to itself by generating spoken language.

For speech recognition we are using the Sphinx software system from Carnegie Mellon University. An overview of an early version of Sphinx is given in [25]. To model the interaction we realized a simple dialog system which is organized in a tree like structure. On the top most level the user can choose a specific task. Depending on the users choice the robot offers further possibilities on the next level of the dialog hierarchy. If the robot was not able to recognize what the human said it can ask the human using a text to speech interface. For speech synthesis we make use of Festival. It was also developed at Carnegie Mellon University and features a simple interface to pass text which is then synthesized as speech. The initial Festival system is documented in [26].

The two modules described above allow for interaction between the robot and a human user. They can be used throughout our complete software system. Because we also enriched the robots internal world representation with semantic information we are providing a natural interface between the robot and its environment. This allows for intelligent robotic applications that can be useful and helpful in human environments.

5 Discussion

In this paper we reported on our activities in RoboCup and showed our approach to design cognitive robots for soccer as well as for service robots. We presented the basic components that allowed for our straightforward transition from soccer to the RoboCup@Home league.

For the soccer robots we showed our approach to behavior specification with ReadyLog using a formal approach based on the situation calculus and Golog. Following ideas of Dylla et al. [8] we use insights taken from human soccer literature for our soccer robots. To be able to transfer the abstract human knowledge more easily we additionally developed a qualitative world model representation. Our kind of qualitative abstraction is also useful for our decision-theoretic planning as it reduces the state space of the planning problem. The hybrid quantitative-qualitative representation together with our underlying programming framework allows for a limited form of reasoning about qualitative world model predicates, which seems expressive enough for most soccer applications.

Then we showed how we adapted our robot system to be able to participate in the new RoboCup@Home league. The basic components of the robot system were designed in such a way that they could be used in the new scenario without

substantial modifications. It was our laser-based localization following a Monte Carlo approach and a very robust collision avoidance and navigation module that provided us with a stable basis to move from the soccer field to the home-like environment. In fact, both these modules work even better in more structured environments and with lower speeds than in the soccer domain. A map building tool which allows for semantic annotations of maps used for localization and navigation was developed. The annotations are available throughout the whole system and especially for the human machine interface. By adapting our ball recognition to other shapes and colors we were also able to detect other objects in the home environment. With the modification we were able to participate in this years' competitions quite successfully (we came in first).

ReadyLog, our variant of Golog, was of very little importance in this years competitions in the @Home league since the tasks did not require much reasoning facilities yet. However, we expect the tasks to become more and more demanding in the future. Still we are very confident that we will be well equipped not only due to the possibility to apply decision-theoretic planning to solve complex tasks.

The main challenges for the future will be to robustly integrate manipulators into our framework in order to be able to physically interact with the environment. Furthermore, the robot's abilities by means of enhanced computer vision will be needed to allow for a cognitive and flexible perception in home-like applications. The qualitative world model developed recently should be employed shortly in order to enhance the human machine interaction in a yet natural way. To sum up we are quite satisfied with the smooth transition from robotic soccer to a service robotics domain. Most of our approaches have proven to be applicable in both domains and the basic principles already did pay off or are likely to do so in the near future.

Acknowledgment
This work was supported by the German National Science Foundation (DFG) in the Priority Program 1125, *Cooperating Teams of Mobile Robots in Dynamic Environments* and a grant by the NRW Ministry of Education and Research (MSWF). We would like to especially thank Tim Niemüller for his contribution to this work. Further, we thank the anonymous reviewers for their comments.

References

1. Ferrein, A., Fritz, C., Lakemeyer, G.: On-line Decision-Theoretic Golog for Unpredictable Domains. In: Proc. of 27th German Conf. on Artificial Intelligence, Springer (2004) 322–336
2. Ferrein, A., Fritz, C., Lakemeyer, G.: Using golog for deliberation and team coordination in robotic soccer. KI Künstliche Intelligenz (1) (2005)
3. McCarthy, J.: Situations, Actions and Causal Laws. Technical report, Stanford University (1963)
4. Strack, A., Ferrein, A., Lakemeyer, G.: Laser-based localization with sparse landmarks. In: Proc. RoboCup 2005 Symposium. (2005)
5. Dellaert, F., Fox, D., Burgard, W., Thrun, S.: Monte Carlo localization for mobile robots. In: Proc. of the Int. Conf. on Robotics and Automation (ICRA). (1999)

6. Moravec, H., Elfes, A.: High resolution maps from wide angular sensors. In: Proc. of the IEEE Int. Conf. On Robotics and Automation (ICRA). (1985) 116–121
7. Chen, T., Chung, K.: An efficient randomized algorithm for detecting circles. Computer Vision and Image Understanding **83**(2) (2001) 172–191
8. 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: RoboCup 2004: Robot World Cup VIII, Springer (2005)
9. Reiter, R.: On knowledge-based programming with sensing in the situation calculus. ACM Transactions on Computational Logic (TOCL) **2**(4) (2001) 433–457
10. Levesque, H.J., Reiter, R., Lesperance, Y., Lin, F., Scherl, R.B.: GOLOG: A logic programming language for dynamic domains. Journal of Logic Programming **31**(1-3) (1997) 59–83
11. Grosskreutz, H., Lakemeyer, G.: cc-Golog – An Action Language with Continous Change. Logic Journal of the IGPL (2002)
12. De Giacomo, G., Lésperance, Y., Levesque, H.J.: ConGolog, A concurrent programming language based on situation calculus. Artificial Intelligence **121**(1–2) (2000) 109–169
13. De Giacomo, G., Levesque, H.: An incremental interpreter for high-level programs with sensing. In Levesque, H.J., Pirri, F., eds.: Logical Foundation for Cognitive Agents: Contributions in Honor of Ray Reiter. Springer, Berlin (1999) 86–102
14. Grosskreutz, H.: Probabilistic projection and belief update in the pgolog framework. In: Proc. 2nd Int. Cognitive Robotics Workshop. (2000) pages 34–41
15. Boutilier, C., Reiter, R., Soutchanski, M., Thrun, S.: Decision-theoretic, high-level agent progr. in the situation calculus. In: Proc. AAAI'00, AAAI Press (2000)
16. Dylla, F., Ferrein, A., Lakemeyer, G.: Acting and deliberating using golog in robotic soccer – a hybrid architecture. In: Proc. CogRob02, AAAI Press (2002)
17. Dylla, F., Ferrein, A., Lakemeyer, G.: Specifying multirobot coordination in ICP-Golog – from simulation towards real robots. In: AOS-4 at IJCAI-03. (2003)
18. Schiffer, S., Ferrein, A., Lakemeyer, G.: Qualitative World Models for Soccer Robots. In: Qualitative Constraint Calculi, Workshop at KI 2006. (2006) 3–14
19. Hernandez, D., Clementini, E., Felice, P.D.: Qualitative distances. In Kuhn, W., Frank, A., eds.: Spatial Information Theory: a theoretical basis for GIS. Number 988 in LNCS. Springer-Verlag (1995) 45–58
20. Clementini, E., Felice, P.D., Hernandez, D.: Qualitative representation of positional information. Artificial Intelligence **95**(2) (1997) 317–356
21. Aurenhammer, F., Klein, R.: Voronoi diagrams. In Sack, J.R., Urrutia, J., eds.: Handbook of Computational Geometry. Elsevier Science Publishers (2000)
22. Randell, D.A., Cui, Z., Cohn, A.: A Spatial Logic Based on Regions and Connection. In: KR'92. Principles of Knowledge Representation and Reasoning: Proc. of the Third Int. Conf. Morgan Kaufmann, San Mateo, California (1992) 165–176
23. Renz, J., Nebel, B.: On the complexity of qualitative spatial reasoning: a maximal tractable fragment of the region connection calculus. Artificial Intelligence **108**(1-2) (1999) 69–123
24. Kuhn, H.: The Hungarian method for the assignment problem. Naval Research Logistics Quarterly **2** (1955) 83–97
25. Huang, X., Alleva, F., Hon, H.W., Hwang, M.Y., Rosenfeld, R.: The SPHINX-II speech recognition system: an overview. Computer Speech and Language **7**(2) (1993) 137–148
26. Black, A.W., Taylor, P.A.: The Festival Speech Synthesis System: System documentation. Technical Report HCRC/TR-83, Human Communciation Research Centre, University of Edinburgh, Scotland, UK (1997)