AllemaniACs@Home 2006 Team Description

Alexander Ferrein, Gerhard Lakemeyer, and Stefan Schiffer

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

Abstract. This paper describes how the scientific advances of the AllemaniACs soccer team can be applied for the 2006 AllemaniACs@Home challenge. In contrast to the team description paper for the AllemaniACs Mid-Size team, we focus on the low-level robot control software which allows us to use the same robots and software in the @Home league as well.

1 Introduction

While in the RoboCup soccer leagues the complexity of the task lies in fast accessing the sensors, quick decision making and cooperation, the challenge in the @Home league is to build a system which enables a robot to robustly and safely navigate through human populated home environments. Since the new RoboCup@Home league focuses on service robotics applications one main challenge is that of human-machine interaction. This year, tasks like "follow a human through the home environment" or "navigate from the fridge to the living room" will be part of the RoboCup@Home competition.

This means for one that the robot must be able to build an internal representation for arbitrary home environments. That is because for the competition the environment that the robot has to operate in is not known in advance. For another, the robot must be able to localize itself in this particular environment and it has to be able to navigate through it safely. This task surely demands for path planning and obstacle avoidance abilities. Our Mid-Size robots which we will also use in the @Home league use a Monte Carlo approach with a laser range finder for localization. Furthermore, they employ an A^*-based collision avoidance algorithm and a path planner which ensures short paths between reachable points in the environment.

The high-level control is based on the language Readylog, a variant of the logic-based language GOLOG [1] which combines explicit agent programming as in imperative languages with the possibility to reasons about actions and their effects. In particular, we are interested in decision-theoretic planning in the Readylog framework which allows to generate optimal plans for complex tasks.

In the sequel we describe our hardware platform in Section 2. We present important aspects of our low-level control system in Section 3, before we sketch the high-level control language Readylog and give an example of a service robotics application of the AllemaniACs team from the 2004 RoboCup Technical Challenge in Section 4.

2 AllemaniACs Robots

The hardware platform we will use in the first RoboCup@Home challenge 2006 are those of the AllemaniACS Mid-Size RoboCup Team. The robot has a size of 39 cm $\times$ 39 cm $\times$ 40 cm (Fig. 2). 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 omnivision camera. Our other sensor is a 360° laser range finder 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 allows allows the use of the hardware platform for soccer playing, but it is also possible to use it for service robotics applications.

Fig. 1: AllemaniACs Robot

3 Low-Level Control

Currently we use the 360° laser range finder as our main sensor for navigation, obstacle avoidance, and localization. In the following we describe the respective modules in greater detail.

3.1 Collision Avoidance and Navigation

The collision avoidance module performs an A* search over an occupancy grid [2] generated from the laser scanner inputs. The robot is positioned in the middle (origin) of the grid. Next, the collision-free path from the current location to a given target point must be calculated. We perform 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 occupied cells 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*

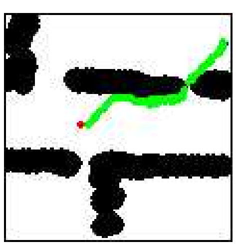
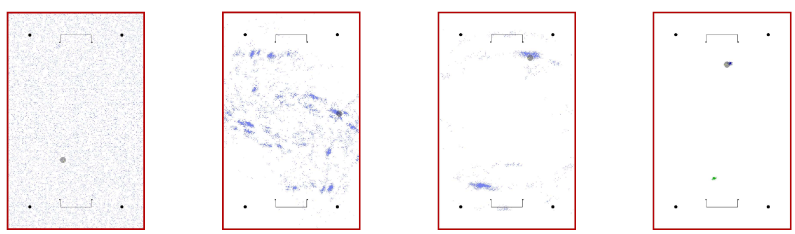


Fig. 2: Path to the target inside a room.

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 A* calculates (depicted in Fig. 2) 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 applying an A* search over the velocity space. This search yields appropriate translational and rotational velocities with which the robot drives to the given target point.

3.2 Localization

Our self-localization uses of the Monte Carlo Localization algorithm [3]. 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 now chooses the most likely hypothesis given the previous estimate, the actual sensor input, the current motor commands, and a map of the environment. Fig. 3 shows a global localization on a soccer field. The blue dots represent one sample of the sample set. In the beginning the robot has no clue about its position and therefore many hypotheses. After driving around and taking new sensor updates the robot's belief about its position condenses to two main hypothesis near the goals. The robot cannot resolved this ambiguity solely with the laser scanner.



(a) (b) (c) (d)

Fig. 3. Simulated global localization on the RoboCup field. The real position of the robot is depicted by the gray circle. Because of the symmetry of the environment model, two clusters have developed.

To be able to localize robustly with the laser range finder we modified the Monte Carlo approach. To be able to integrate a whole sweep from the LRF we use a heuristic perception model. With this we are able to localize with high accuracy in the RoboCup environment. The method is presented in detail in [4]. Our approach, which was inspired by the RoboCup setting, works also very well for indoor navigation even in large environments.

3.3 Object Classification

For localizing the robot in the environment we build an occupancy grid map of the environment. With this map, we are able to determine dynamic obstacles in the environment when new laser readings arrive: every object which is not represented in the map is assumed to be a dynamic obstacle. To be able to distinguish between different dynamic objects, we use the laser signature of the objects. In the soccer setting we are able to distinguish between our own robots and opponents, and even humans can be told apart. Though, the only important information there is whether the object is a teammate or an opponent obstacle. Therefore, our heuristic for classification is very rough at the moment.

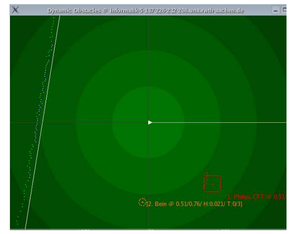


Fig. 4: Identified Objects.

Fig. 4 shows a screenshot of the object classification. The red box shows a classified Philips robot, the circle shows scan points hat were classified as a human leg.

4 Readylog

For specifying our high-level control we use a variant of the logic-based high-level agent programming language GOLOG [1]. GOLOG is a language based on the situation calculus [5]. Over the past years many extensions like dealing with concurrency, exogenous and sensing action, a continuous changing world and probabilistic projections (simulation) [6,7,8] made GOLOG an expressive robot programming language. We integrated those features in our READYLOG interpreter [9]. For the decision making, we further integrated a planning module into GOLOG which chooses the best action to perform by solving a Markov Decision Process (MDP) (we refer to [10] for reading on MDP and to [11] on integrating MDPs into GOLOG).

The READYLOG language features standard constructs like action sequences, procedures, conditionals, loops, but also less standard constructs like nondeterministic choice of actions, probabilistic projections into the future (simulations), or decision-theoretic planning are supported. For examples of how a multi-agent plans for the robotic soccer domain can be formulated in READYLOG we refer to [12,9]. For further extensions of the READYLOG interpreter we refer to [13,14].

Recently, we finished our work on a qualitative abstraction of the world model for the MID-SIZE domain [15,16]. The qualitative world model is integrated in the READYLOG language and used for abstract planning. The qualitative world model provides abstractions for positional information such as *left* or *right* as well as higher-level concepts like that of reachability which is fundamental in soccer. The qualitative spatial data provided by this world model are based on human cognition. Thus, they render useful especially when it comes to human-machine interaction since the robot can handle information which originate from human language more easily.

(a) Part of the RoboCup 2004 Site in Lisbon

(b) The navigating robot

Fig. 5. Technical Challenge 2004

Readylog Example: RoboCup 2004 Technical Challenge

In the following we give a brief example of READYLOG to show that our robot control software is well suited for service robotics applications. In the ROBOCUP MID-SIZE Technical Challenge 2004 we presented a service robotics application: the robot was to drive autonomously to one particular soccer field which was chosen by one of the referees. The robot calculated the shortest way to the field, announcing historic sights of the exhibition hall like the stand of the fields or the pillars of the hall on the way.

Fig. 5(a) shows the occupancy map of the rear part of the exhibition hall. In the upper part you could detect two of the MID-SIZE fields.

```
proc( pathPlan( Goal, H ),
      solve( while( not( mapNode = Goal ),
                    pickBest( child, childrenOf( mapNode ),
                              gotoMapNode( child ))),
             H, beAt( Goal ))).

function( beAt( Goal ), V, if( mapNode = Goal, V = 100, V = 0 )).

proc( gotoMapNode( Node ),
      [goto_global(ownNumber, nodeCoord( Node ), 1),
       playSound( Node )]).
```

Fig. 6. The READYLOG program for the 2004 challenge.

The high-level control program is shown in Fig. 6. The parameter *Goal* specifies the target location. Inside the *solve* statement the planning takes place: as long as the robot has not reached the target, it chooses the best neighbor to the current node in its topological map (which is not shown here) applying the *pickBest* statement. The reward function we use here is the *beAt* function, which gives a high reward at the target position and 0 otherwise. The cost function for an action which is also not presented here is defined as the Euclidean distance between nodes in the topological map. Thus, the robot finds the shortest sequence of nodes to the target position. At each node the robot calls the *gotoMapNode* procedure and plays a sound file to announce the exhibit. Not without pride, we want to mention that we won the silver medal in the 2004 Technical Challenge with this demonstration .

References

1. 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** (1997) 59–83
2. Moravec, H., Elfes, A.: High resolution maps from wide angular sensors. In: Proc. of the IEEE International Conference On Robotics and Automation (ICRA). (1985) 116–121
3. 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)
4. Strack, A., Ferrein, A., Lakemeyer, G.: Laser-based localization with sparse landmarks. Proc. RoboCup Symposium 2005 (2005)
5. Reiter, R.: Knowledge in Action: Logical Foundations for Specifying and Implementing Dynamical Systems. MIT Press (2001)
6. Giacomo, G.D., Lésperance, Y., Levesque, H.J.: ConGolog, A concurrent programming language based on situation calculus. Artificial Intelligence **121** (2000)
7. Grosskreutz, H., Lakemeyer, G.: cc-Golog: Towards more realistic logic-based robot controllers. In: Proc. AAAI'00, AAAI Press (2000) 476–482
8. Grosskreutz, H.: Probabilistic projection and belief update in the pGolog framework. In: Second International Cognitive Robotics Workshop. (2000)
9. Dylla, F., Ferrein, A., Lakemeyer, G.: Specifying multirobot coordination in ICP-Golog – from simulation towards real robots. In: Proc. Physical Agents WS on IJCAI'03. (2003)
10. Puterman, M.: Markov Decision Processes: Discrete Stochastic Dynamic Programming. Wiley (1994)
11. Boutilier, C., Reiter, R., Soutchanski, M., Thrun, S.: Decision-theoretic, high-level agent programming in the situation calculus. In: AAAI'2000. (2000)
12. Ferrein, A., Fritz, C., Lakemeyer, G.: Using golog for deliberation and team coordination in robotic soccer. KI (2005)
13. Ferrein, A., Fritz, C., Lakemeyer, G.: On-line decision-theoretic golog for unpredictable domains. In: Proc. of 27th German Conference on AI. (2004)
14. Ferrein, A., Fritz, C., Lakemeyer, G.: Extending DTGolog with Options. In: Proc. IJCAI'03. (2003)
15. Schiffer, S., Ferrein, A.: A Qualitative World Model for Autonomous Soccer Agents. (2006) submitted.
16. Schiffer, S.: A Qualitative Worldmodel for Autonomous Soccer Agents in the ReadyLog Framework. Master's thesis, Knowledge-based Systems Group, Computer Science Department, RWTH Aachen University, Aachen Germany (2005)