Simulation League, Team *RobologKoblenz99*, pages 116–120
http://www.ep.liu.se/ea/cis/1999/007/25/

RoboLog Koblenz: Spatial Agents Implemented in a Logical Expressible Language

RobologKoblenz99

Frieder Stolzenburg, Oliver Obst, Jan Murray, Björn Bremer

FS, OO, JM, BB: Universität Koblenz

Abstract. *In this paper, we present a multi-layered architecture for spatial and temporal agents. The focus is laid on the declarativity of the approach, which makes agent scripts expressive and well understandable. They can be realized as (constraint) logic programs. The logical description language is able to express actions or plans for one and more autonomous and cooperating agents for the RoboCup (Simulator League). The system architecture hosts constraint technology for qualitative spatial reasoning, but quantitative data is taken into account, too. The basic (hardware) layer processes the agent's sensor information. An interface transfers this low-level data into a logical representation. It provides facilities to access the preprocessed data and supplies several basic skills. The second layer performs (qualitative) spatial reasoning. On top of this, the third layer enables more complex skills such as passing, offside-detection etc. At last, the fourth layer establishes acting as a team both by emergent and explicit cooperation. Logic and deduction provide a clean means to specify and also to implement teamwork behavior.*

1 Introduction

The major goals of the RoboLog project, undertaken at the University of Koblenz, Germany, are the following:

- A flexible, modular system architecture should be established, meeting the various requirements for RoboCup agents. For example, on the one hand, agents have to be able to react in real-time. But on the other hand, it is also desirable that more complex behavior of agents can be programmed easily in a declarative manner.
- It should be possible to handle different representation formats of knowledge about the environment. Information may be quantitative or qualitative, pictorial or propositional in nature. Therefore, we propose a deductive framework, that is expressible in plain first-order logic (possibly plus constraint technology components), that integrates axiomatic approaches in geometry, spatial constraint theories and numerical sensor data.

- Agents not only should be able to act autonomously on their own, but also to cooperate with other agents. For this, we develop a multi-agent script language for the specification of collective actions or intended plans that are applicable in a certain situation. These scripts can be translated into logic programs in a straightforward manner.

2 Basic Abilities and Actions (Layer 1)

In the following, we briefly discuss some aspects of each layer in our system architecture. The basic layer hosts reactive behavior. It is implemented in the RoboLog Prolog extension [3]. This extension is an enhanced RoboCup SoccerServer interface for ECLiPSe-Prolog [2]. Time critical and computational expensive tasks are handled within the RoboLog module, as well as the exchange of data. The module provides the atomic SoccerServer commands and some more complex actions.

Following the lines of [4], we distinguish two classes of predicates: ACTIONS a and PERCEPTIONS p. When executed successfully, a perception predicate p returns the requested data. We will assume, that this data is quantitative, i.e. some arguments of the predicate are real numbers. The main matter of an action a is its side-effect, i.e. the performed action. Nevertheless, an action predicate (except the primitive actions of the SoccerServer) also is assigned a truth value, depending on the success or failure of the action. In summary, the RoboLog interface provides the following functionality:

- For each agent, it requests the sensor data from the SoccerServer. By this, the agents' knowledge bases are updated periodically. Each agent stores information about objects it has seen within the last 100 simulation time steps. So we can think of it as the agent's memory or recollection.
- This low-level data is processed in such a way that more complex and more precise information becomes available, such as global position information or direct relations between objects with or without reference to the actual agent.
- The passing of time can be modeled in several ways with RoboLog. It provides various means for creating snapshots of the world and defining an event-driven calculus upon them, keeping track of the actual simulator step time or just ignoring real-time at all.
- Last but not least, Prolog predicates are provided that can be used to request the current status of sensor information on demand. The data should be synchronized with the SoccerServer, before an agent's action is initiated.

An important piece of information for an agent is to know its own position. Therefore, the RoboLog system provides an extensive library that makes precise object localization possible. The whole procedure implemented in the RoboLog kernel is able to work even when only little or inconsistent information is given. In particular, we employ the method for mobile robot localization using landmarks stated in [1].

In our system, the following basic skills (among others) are implemented:

- The agents can search for the ball, taking into account their knowledge about the last time the ball was seen.

- Dashing and kicking to a certain position, regarding the agent's condition and avoiding obstacles is possible and (based upon these skills) also dribbling.
- Extrapolating the ball trajectory to a given time in the future enables the agents to intercept opponent passes and block shots.

3 Qualitative Spatial Reasoning (Layer 2)

What we need in order to identify situations is the abstraction of quantitative data onto a qualitative level. Therefore, we have another class of predicates—in addition to the classes mentioned in Sect. 2—, namely QUALITIES q. For example, concerning the distance of an agent to the ball in the RoboCup scenario only a few (qualitative) aspects are interesting. Thus, in RoboLog we only distinguish few distances: *close* (the ball is in the kickable area), *near* (the agent is able to detect much detail by its sensors), *short* (maximal shooting distance), *far away* (sensor data become unreliable from this distance), *remote* (out of reach). Quantitative distance intervals can be mapped to qualities. Concerning the other direction, chosen plan schemes must be instantiated with quantitative data for the actual execution.

4 Higher Abilities (Layers 3 and 4)

Many tasks require deeper reasoning, which can be expressed within a Belief-Desire-Intention (BDI) agent architecture. In our context, a BELIEF b is a qualitative predicate q, its negation $\neg q$ or a conjunction of beliefs $b_1 \wedge b_2$. A GOAL g is either an *achievement* goal $!q$ or a *test* goal $?q$, where q is a qualitative predicate. A DESIRE (or event) d is a goal or an action, indexed by a list of agents—the actors—, which must satisfy the desire by performing some actions.

Now we can build rules for a certain SITUATION in form of scripts, written $d : b - i$, where d is a desire, b is a belief (identifying the precondition of the situation), and i is the INTENTION (or, strictly speaking, the intended plan). The intended plan is an acyclic graph of desires with a designated start node. Its edges are labeled with actors which must be a subset of the actors in d. Edges outgoing from test goals are labeled in addition with *yes* or *no* and possibly a time-out delay. Consider now all possible subgraphs wrt. edges for a certain actor. It is required that this is a tree with the start node as root, where binary branching is only allowed after test nodes. These subgraphs represent the ROLE for the respective actor. An achievement goal has to be performed actively by the indexed actors, while non-actors wait for the achievement until a certain time-limit. If the time-limit is exceeded or an external interruption occurs (e.g. a referee message in the RoboCup scenario), then the agent has to return to a *default plan*, which must be applicable without precondition.

Let us now consider an example for a collective action of agents, namely double passing. There are two actors in this situation: actor 1 kicks the ball to actor 2, then actor 1 runs towards the goal, and expects a pass from actor 2. The respective rule can be expressed as shown in Fig. 1.

We may distinguish several types of plans: basic plans with only one actor and complex plans where there are more than one actors. The former plans implement higher abilities (layer 3), while the latter realize teamwork (layer 4). For each

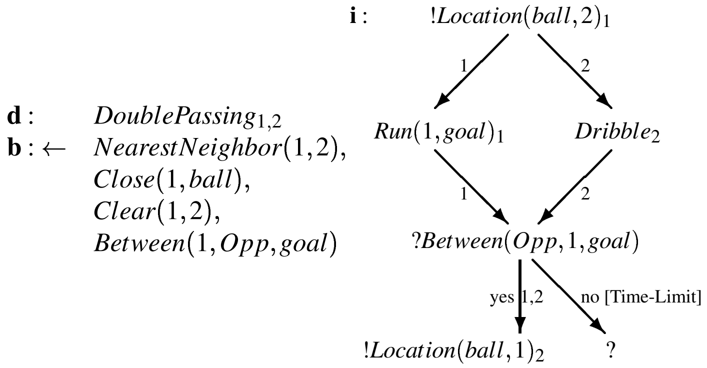

Figure 1: Double Passing Script.

situation and for each role in it, a BDI script can be translated directly into a logic program rule, possibly with concurrent constraints (belief conditions):

$$d \leftarrow b \wedge i$$

5 Summary

We presented a logical description language for multi-agent systems, following the lines of [4]. The implementation language can be understood as a generalization of CLP. Both quantitative and qualitative spatial reasoning can be built-in.

The RoboLog system provides a clean means for programming soccer agents declaratively. The RoboLog Koblenz players were implemented by a team of 3 to 5 people. We conducted several test games with different scores on our local network—a 100 MBit Ethernet. The results of some successful games are shown in Fig. 2.

RoboLog Koblenz (on 3 Pentium II Linux-PCs and 2 Sun Sparc Ultra-1, 143 MHz, all 64 MB main memory)	*Linköping Lizards* (on 1 Sun Ultra-Enterprise with 14 336 MHz processors, 3 GB main memory)	**6:1**
RoboLog Koblenz (on 1 Sun Ultra-Enterprise with 14 336 MHz processors, 3 GB main memory)	*AT Humboldt 97* (on 1 Sun Sparc 5, 110 MHz, 64 MB main memory)	**2:0**

Figure 2: Successful soccer simulation games.

But there are still some problems to solve. Apart from several small questions, we identified two main problems, which often have a huge effect on the players' behavior:

- The execution of a collective action often fails because of the imperfect implementation of the low-level facilities. Although all active agents recognize the situation and their special role correctly, the intended plan may fail, e.g. because a pass from agent 1 does not reach agent 2. To minimize such faults, we do not only improve the low-level skills implemented so far, but also work on adapting the skills of the *CMUnited Simulator Team*, especially the predictive locally optimal skills (PLOS) [5], to RoboLog.

- Another point is that only few situations like double passing are implemented yet. So the agents very frequently have to stick to their default plans, because situations arise which are unclear for the agents. But this problem can be solved by axiomatizing more situations and providing the according scripts to the agents.

Further work should concentrate on the real-time requirements in exceptional situations and the concurrency of different mechanisms for information acquisition. One area of research is how far logical mechanisms can be used within the lower levels of our approach. Deduction could be used to build a more complete view of the agent's world. Each time an agent gets new information, a set of logical rules rebuilds the agent's spatial database. The application of these techniques to real robots is one of the next steps of our research activities. In addition, the robustness of the decision process can be improved by means of defeasible reasoning. The specification of a communication language and the use of any-time reasoning formalisms should also be investigated.

References

[1] M. Betke and L. Gurvits. Mobile robot localization using landmarks. *IEEE Transactions on Robotics and Automation*, 13(2):251–263, Apr. 1997.

[2] International Computers Limited and IC-Parc. *ECLiPSe User Manual / Extensions User Manual – Release 4.0*, 1998. Two volumes.

[3] O. Obst. *RoboLog – An ECLiPSe-Prolog SoccerServer interface: Users manual*, March 1998.

[4] A. S. Rao. AgentsSpeak(L): BDI agents speak out in a logical computable language. In W. van de Velde and J. W. Perrame, editors, *Agents Breaking Away – 7th European Workshop on Modelling Autonomous Agents in a Multi-Agent World*, LNAI 1038, pages 42–55, Berlin, Heidelberg, New York, 1996. Springer.

[5] P. Stone, M. Veloso, and P. Riley. The CMUnited-98 champion simulator team. In M. Asada and H. Kitano, editors, *RoboCup–98: Robot Soccer World-Cup II*, LNAI. Springer, Berlin, Heidelberg, New York, 1999. To appear.