AT Humboldt in RoboCup-98 (Team description)

Pascal Gugenberger, Jan Wendler, Kay Schröter, and Hans-Dieter Burkhard

Institute of Informatics
Humboldt University Berlin, D-10099 Berlin, Germany **
email: muellerg/wendler/hdb/kschroet@informatik.hu-berlin.de
WWW: http://www.ki.informatik.hu-berlin.de

Abstract. The paper describes the scientific goals of the virtual soccer team "AT Humboldt 98", which became vice champion in RoboCup-98 in Paris. It is the successor of the world champion "AT Humboldt" from RoboCup-97 in Nagoya.

1 Introduction

The virtual soccer teams "AT Humboldt 97" and "AT Humboldt 98" (AT stands for "Agent Team") are implemented by our AI group at the Institute of Informatics at the Humboldt University Berlin. The work is done by groups of students as practical exercises for the advanced course "Modern methods in AI" during summer semester. A core group of three students maintains the coordination and the programs.

The new program "AT Humboldt 98" [1] is based on the architectural concepts of the successful program from RoboCup-97 in Nagoya. We decided to make a re-implementation for more rigid structuring. The new team has more skills, more complex deliberation processes, and new facilities for on-line learning.

The team from Nagoya took part in RoboCup-98 under the name "AT Humboldt 97". The idea behind its nomination was the possibility to compare the development in virtual soccer from Nagoya to Paris. We therefore didn't want to change the program, but this was not possible because of the new rules (changes in the soccer server). We tried to make only as few changes as possible. Necessary changes concerned the new parameters. At the end, "AT Humboldt 97" still was one of the top 16 teams in Paris. The main handicap arose from the new offside rule: There was no feature in "AT Humboldt 97" to avoid offside. Teams exploiting the offside rule could easily stop the offense of AT Humboldt 97.

2 Scientific Goals

We are interested in virtual soccer for the development and the evaluation of our research topics in artificial intelligence which concern the fields of

** The work was partly sponsored by infopark online service

[1] the sources are available from our web pages: http://www.ki.informatik.hu-berlin.de/RoboCup/RoboCup98/index_e.html

M. Asada and H. Kitano (Eds.): RoboCup-98, LNAI 1604, pp. 358-363, 1999.

- Agent oriented techniques (AOT),
- Multi-Agent Systems (MAS),
- Case Based Reasoning (CBR).

Thus many aspects of our soccer program are heavily influenced by these fields, but it is important not to consider these fields in isolation: to create our soccer agents, we also needed a lot of contributions from other fields of computer science (e.g. programming techniques, synchronization, concurrency) and from mathematics. Thereby we gain deeper insights for integration of AI techniques in software development. This aspect is especially important for the education of our students.

2.1 Agent oriented techniques (AOT)

Our understanding of AOT is closely related to new developments and new requirements in software technologies, which are driven by new expectations to programs and intelligent systems by a broad audience (not limited to computer scientists: this distinguishes AOT from e.g. object oriented techniques). Characteristic aspects of related agent-programs (we do not want to give one more definition of "agents") are e.g. autonomy, cooperation, rational behavior and mental qualities (the programs use "knowledge" for their "decisions" and they can deal with "orders" by their users), etc. AOT should support related functionalities. Up to now there is no common understanding of what agent oriented techniques may be (but remember that object oriented programming needed 20 years of development).

RoboCup is an ideal environment for testing appropriate structures and programming techniques. Our agent architecture uses a mental deliberation structure which is best described by a belief-desire-intention architecture (BDI) [3]. Distinct from other (e.g. logically motivated) approaches our approach is closely related to procedural thinking, and we use object oriented programming (C++, Java) for the implementation.

2.2 Multi Agent Systems (MAS)

Our interest in RoboCup for MAS concerns the cooperation between agents in the presence of opponents. Special emphasis is given to emergent cooperation: How can agents cooperate only by observing each other (or better to say: how can we implement cooperative behavior by using our knowledge about the programmed behavior of our agents). Social behavior results from common individual rules. In the future we will try to compare emergent cooperation with cooperation by explicit communication of intentions. Experiments with communication of world state information did not lead to significant improvements.

2.3 Case Based Reasoning (CBR)

Our understanding of CBR [2] means learning from former experiences (cases) especially for situations where we have not enough information to induce rules.

Successful CBR needs efficient case memories which permit the retrieval ("reminding") of old cases in short time.

RoboCup offers a lot of scenarios for learning from experiences. We distinguish between "off-line learning " (training), where we can make a lot of experiments for collecting cases, and "on-line learning " during the matches where we can collect only few cases in order to learn the opponents tactics and skills.

Cases from "off-line learning " can be used to extract rules for behavior and to tune parameters.

3 The Architecture of "AT Humboldt 98"

Figure 1 depicts the overall structure of "AT Humboldt 98". The arrows indicate the data-flow. The sensors parse the information coming from the Soccer Server and cause their integration into the internal world model. The deliberation component decides what to do based on the data in the world model. As a result of its deliberation it creates a plan of atomic actions using the available skills, which also make use of the world model. The deliberator hands the plan over to the effectors that manage its execution and inform the world model about the actions the agent has sent to the server.

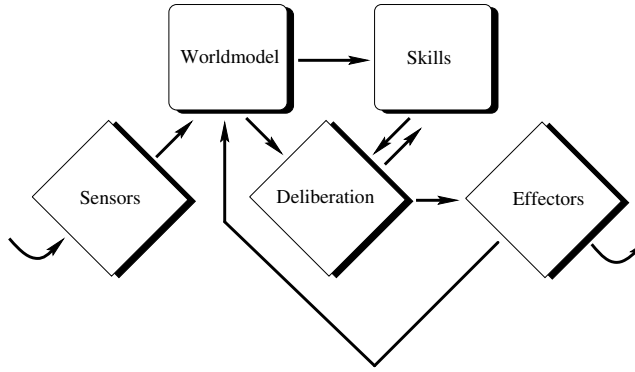

Fig. 1. Overall agent architecture

3.1 Skills

Skills enable the deliberator to work on a more abstract level by encapsulating "subconscious" tasks like running, kicking the ball or dribbling. They decompose intentions like "Move to position x, y" into a series of atomic actions. Often it is not necessary to compute the complete series, since new incoming sensor information frequently changes world model so significantly that the part of the plan that hasn't been executed yet has to be adapted to this information or even completely recalculated.

One of the most important skills is the kicking of the ball. The implemented skill accelerates the ball in a given direction, trying to achieve a given final speed.

If this speed cannot be reached, the skill tries to maximize it. If the player is in the way of the ball, the skill moves the ball around the player [2].

Another skill is running to a given position. It tries to compute a plan that enables the player to reach this position as fast as possible. At the same time it ensures that the stamina of the player does not decrease below a certain threshold depending on the current intention. Obstacles on the path to the target position are avoided. The player can also run backwards with this skill.

The third important skill is dribbling. It allows the player to move forward while keeping the ball within the player's control radius. The skill tries to keep the players body between the ball and the closest opponent. Just like the previous skill dribbling doesn't let the player's stamina drop below a given threshold.

3.2 World Model (belief)

All information concerning the outside world is a part of the belief. New sensor information updates the world model. Parameters of objects outside of the visual range are estimated by simulation. The world model can perform simulations into the future and estimate e.g. shortest paths for intercepting the ball.

Data that belongs to the same simulation step is stored together in an object called situation. It contains the representation of the players and the ball. Speed and position values are stored together with reliability values that indicate how old the underlying sensor information is. The world model also contains internal knowledge about the base position of the player, which usually changes during the course of a game, and the role of the player, which remains constant.

3.3 Deliberation (desire, intention, plan)

Deliberation starts whenever an update of the world model is completed or the current plan is finished. First it looks for an existing plan and evaluates the conditions for continuation. If it does not decide to continue this plan, then it evaluates all options by calculating a rough estimate of their expected utilities. Options with a utility above a certain limit are chosen as desires, i.e. as candidates for a new intention.

During the second phase an intention is chosen out of the current desires. Starting with the highest scored desire we check if the desire is feasible, i.e. if a related plan can be computed. If so, this desire becomes the agent's intention. Otherwise the remaining desires are examined.

Currently we are using only single intentions. Further conditions are modelled as *constraints*. The utilities of obeying the constraints are regarded while computing the utility estimates for the options.

After the commitment to an intention, a new plan is computed based on related skills. As a result, a sequence of actions is given to the effector module which sends them to the soccer server while maintaining synchronization. If

[2] actually, our kick implementation was not completely finished in AT Humboldt 98

an intention resp. plan is later dropped by the deliberator, then a new plan overwrites the old one in the effector.

The stability of intentions/plans is always in conflict with the re-deliberation and adaptation to new situations. We use the following procedures: There is a specification for the current intention under which conditions it must not be cancelled. This condition is checked whenever the deliberator starts. Furthermore, we compare a new chosen intention with the old one at the end of the deliberation process.

4 Learning techniques

We distinguish between off-line learning ("training") and on-line learning (adaptation during the matches). Our approaches are in an experimental stage.

The success of skills depends on appropriate choices of the consecutive actions. Thereby, learning should not concern a single action, but the whole sequence of actions. Learning for skills can be performed as off-line learning. We have made some experiments for learning, but up to now the skills have been hand coded according to an analysis of situations.

Our experiments concern the training of the ball-shooting skill which is performed by several kick commands. Data is collected using an automatized coach mode. This data is analyzed in order to find optimal parameters and hopefully to find rules for computing the optimal parameters.

Another experiment concerns the choice of good base positions using on-line learning. Good positions depend strongly on the opponents. We have recorded positions during a game in a raw grid of the field and then adapted the player positions to that knowledge. We have used this strategy in some of our matches in Paris, but we have not really been satisfied by the results.

Choices in the deliberator depend on several parameters (especially for utility calculations). These parameters may be tuned in a general way (off-line learning) and regarding the behavior of opponents during a match (on-line learning). We have experimented with CBR for the deliberation concerning dashing to a good position in a concrete situation. Cases contain information about positions, expected behavior, and stamina. Our strategy could be used for both off-line learning and on-line learning, respectively. First experiences are reported in [4].

5 Implementation Issues

The re-implementation of our programs for "AT Humboldt 98" follows a consequent object oriented design methodology. We use C++ for the implementation. Java would have been an alternate choice under the aspect of software technologies, but was ruled out because of of the slow implementation of the Java Virtual Machine on our machines. AOT is used for the structure (architecture) of our programs as described above.

To support the concurrent development we use the freely available source code management system CVS [5] and the documentation system doc++ [6].

6 The Development of the "AT Humboldt" Programs

We started in early 1997 with the implementation of a first prototype (March 1997). The design and implementation of a soccer agent was the topic of practical exercises for the advanced course "Modern methods in AI" during summer semester 1997 (April – July). Different architectures and learning concepts have been discussed and partially implemented in C++ and Java. The best concepts were chosen for the final implementation of the program of "AT Humboldt" for RoboCup 97 in Nagoya. Because of performance we decided to use C++.

The first running version was built in the beginning of August by a group of 8 students. The structuring according to AOT allowed significant improvements in the remaining short period of time. The usage of learning (especially CBR) could not be realized. The reasons for success in Nagoya could be seen in the efficient skills and the emergent cooperation based on simple principles.

The re-implementation for "AT Humboldt 98" was the work of a core group of three students. The extensions and new features were again the topic of the practical exercises for the course "Modern methods in AI" during summer semester, although we had serious timing problems because RoboCup-98 was scheduled more than a month earlier than in 1997.

Our experiences with RoboCup under educational aspects are very promising: Students work in a larger project which they have to organize by themselves. The project includes the development of own concepts (at the beginning, it was completely open, which concepts would be useful). Successful implementation needs the integration of a lot of different concepts not restricted to AI.

Acknowledgment

The authors want to give their special thanks to all students involved in our RoboCup projects and to the sponsor infopark online service.

References

1. Burkhard, H. D., Hannebauer ,M., Wendler, J. : AT Humboldt — Development, Practice and Theory. RoboCup-97: Robot Soccer World Cup I, Springer 1998, 357–372.
2. Lenz, M., Bartsch-Spörl, B., Burkhard, H. D., Wess, S.: Case Based Reasoning Technology. From Foundations to Applications. LNAI 1400, Springer 1998.
3. Rao, A. S., Georgeff, M. P. : BDI agents: From theory to practice. In V. Lesser, editor, *Proc. of the First Int. Conf. on Multi-Agent Systems (ICMAS-95))*, pages 312–319. MIT-Press, 1995.
4. Wendler, J., Lenz, M.: CBR for Dynamic Situation Assessment in an Agent-Oriented Setting. In D. Aha and J. J Daniels, editors, *Proc. AAAI-98 Workshop on Case-Based Reasoning Integrations, Madison, USA, 1998.*
5. CVS: http://www.loria.fr/ molli/cvs-index.html
6. DOC++: http://www.zib.de/Visual/software/doc++/index.html