

AT Humboldt in RoboCup-99

ATHumboldt99

Hans-Dieter Burkhard, Helmut Myritz, Gerd Sander, Thomas Meinert

HB, HM, GS, TM: Humboldt University Berlin

Abstract. *The paper describes the architecture and the scientific goals of the virtual soccer team “AT Humboldt 99”, which is the successor of vice champion “AT Humboldt 98” from RoboCup-98 in Paris and world champion “AT Humboldt 97” from RoboCup-97 in Nagoya. Scientific goals are the development of agent-oriented techniques and learning methods.*

1 Introduction

The virtual soccer teams “AT Humboldt” (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 in practical exercises during the summer semester. A core group of up to three students maintains the coordination and the programs.

Besides the experiments with AI methods, the project is also a challenge for software development by changing teams: The source code of AT Humboldt 98 has about 25 000 lines of code [4]. It is a non trivial task to maintain the introduction of new ideas during extremely short time intervals (for Paris we had only two month).

When writing this description (May 99), some of the new features of AT Humboldt 99 are still ideas. AT Humboldt 99 will be an extension of AT Humboldt 98 with additional and (hopefully) improved skills, new options, and a larger planning horizon, respectively. Again, the work is done by small teams of students. Most of them work the first time for RoboCup.

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

- 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) [8]. 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.

In the BDI-approach, agents maintain a model of their world which is called belief (because it might not be true knowledge). The way from belief to actions is guided by the desires of the agent. Bratman has argued that intentions are neither desires nor beliefs, but an additional independent mental category. Intentions are considered as (partial) plans for the achievement of goals by appropriate actions. Commitment to intentions is needed which has impact on the rational usage of resources: The (relative) stability of committed intentions prevents overload in deliberation and useless plan changes, and it serves trustworthiness in cooperation. This aspect becomes more and more important as we are implementing long term intentions in the future.

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 “AT Humboldt 99” we will compare emergent cooperation with cooperation by explicit planning of long term intentions. Stability of intentions is a fundamental for successful cooperation.

2.3 Case Based Reasoning (CBR)

Our understanding of CBR [6] 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.

3 Components of AT Humboldt 99

The agents of the AT Humboldt teams [1; ?; ?; ?; ?] can use different *skills* like GO-TO-POSITION, KICK, DRIBBLING, etc. . Each skill can be considered as a partial plan or a parameterized procedure. It consists of a sequence of atomic actions which are specified in detail according to the underlying situation.

The *belief* component models the belief of the agent about the environment according to sensor information and simulation for moving objects. It provides simulation and evaluation of speculative future developments for possible actions.

The *desire* component evaluates “options” (potential desires) according to the beliefs. Typical options are “intercept ball”, “goal-kick”, “dribbling”. The commitment for a desire is based on the expected success using certain heuristics and simulation methods.

The *intention* component specifies the best plan according to a committed desire. It uses the predefined skills. The intention component parameterizes these plans according to the expectation of optimal success.

After deliberation is complete, a sequence of atomic actions is given to the *execution* component which is responsible for the synchronization with the Soccer Server.

The decision processes depend on the role of the players. The roles define special preferences which concern the direction of kicks and the frequency of dribbling, respectively. This emerges in special forms of cooperative play which gives preferences to a more aggressive play over the wingers.

4 Deliberation (desire, intention, plan)

BDI reasoning means to us the rational choice of promising options to become desires and intentions. This process should show the following properties, which are related to bounded rationality inside the agent as well as to software technological requirements:

Time-boundedness : In a real-time environment the reasoning process is bounded by time restrictions. Either the environment may enforce a

timely decision (e.g. in applications with security demands) or late decisions may lead to suboptimal behavior (i.e. missing an opportunity to act).

Distinction of Control and Knowledge : The control of the reasoning process itself should be generic, such that it is independent from the domain-specific options and remains flexible.

Independence of Options : Alternating, deleting or adding an option should influence the reasoning process and other options only marginally or even not at all. This property could be called *scalability*.

The main idea, which guarantees the fulfillment of the above mentioned demands, is the decomposition of the reasoning process into modular heuristic options. In our model, every option implements a standardized interface, which defines an efficient utility estimation, an attainability predicate, a layered planner and a continuation enforcement predicate. We distinguish between implicate persistence (reconsideration of plans is forced to stabilize an existing plan) and explicit persistence (replanning is delayed by explicitly determined time intervals for reconsiderations). More details are given in [5].

We are still experimenting with a new feature which would change the deliberation process gravely. Under the abstract "STRATEGICPLAN" we are looking for the possibilities of a new stage in the deliberation process. The aim is to build a structure in which the agent can have long lasting intentions of high complexity. These intentions are based on the options we discussed earlier.

For example our agent would decide in a given moment to go to a fixed position, then waiting for an other player to pass the ball towards his new position and finally kick the ball to the goal. This queue of actions is the result of a couple of intentions and plans which are managed by the STRATEGICPLAN. The calculation of the future plans is done in time slots the agent can have between his normal deliberation processes. Only the intentions have to be build in the initial phase of the current strategic plan. Agents can communicate their strategic plans to their teammates.

5 Learning techniques

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

We are experimenting with the training of the ball-shooting skill. Therefore we collect data using different parameters in different situations in an automatized coach mode. This data is analyzed in order to find optimal parameters and hopefully to find computational rules for determining the optimal parameters according to different situations. Analysis can be done by special programs ("coach") outside of the player programs. After having some results, the next phase of training is to be done with improved parameters for further fine tuning.

Another trainable skill is the interception of a moving ball. It is important in the deliberation process, which players can intercept the ball first (e.g.

for the decision if the agent should run to the ball, or for the decision concerning a pass to a team mate, respectively). Related computations are necessary several times during the deliberation process. Because of ball decay, the optimal point for fast interception is difficult to compute. In the first implementation, we used simulation of ball and players, which was extremely time consuming resulting in too long deliberation times. Now we have replaced the simulation by a large table of interception points, which was generated by examples. Later it may be reduced by learning techniques.

A further example of an trainable skill is the way how players could run with the ball to a given position. This skill we call DRIBBLE. Because of the ball decay and the player movement it is difficult to compute the low level actions. In the first implementation, we used simulation of ball and players, which again was extremely time consuming. When writing this description we are working to replace the simulation by a scheme for the approximation of the optimal combination of kick and dash comands, which was obtained from a set of generated examples.

In general terms, we consider off-line learning as the improvement of action sequences which are rawly described in the beginning and more and more improved by experience (as in human training, where the general course of actions is taught “mentally”, but the fine tuning is performed by doing).

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.

In another experiment we have collected situations according to successful movements of players. The neighborhood of the player was divided into segments, which were marked according to the players. In a new situation, the player retrieves similar cases from the case base and makes a move which is suggested by the previous situations. First experiences are reported in [9].

6 Implementation Issues

The Implementation of our programs for “AT Humboldt 99” is based on the programs of “AT Humboldt 98” [3; 4; 7]. It follows a consequent object oriented design methodology using C++.

Traditionally, BDI approaches are embedded in frameworks of multi-modal temporal logics. This makes them unattractive for programming in procedural or object-oriented environments. We found that object-oriented programming is best suitable for programming our soccer agents, and we have chosen C++ for performance reasons. Java could have been an alternate choice under the aspect of software technologies, but was ruled out because of the slow implementation of the Java Virtual Machine on our machines. The BDI approach gives us the structure for our agent architecture.

In our approach, procedures/methods are considered as the basic behavior for agents similar to traditional programs. Actually, the skills of our agents are procedures. But the call of procedures in traditional programs follows a control flow which is strictly chosen already by the programmer. This is not

possible in highly dynamic situations: The programmer can not explicitly define what the agent has to do in all possible situations. The agent has to analyze the situation and to decide autonomously by related criteria what to do. The programmer can implement only the methods for such decisions (to give an example: the programmer of a chess program does not know which move his program will do in a concrete situation). We found, that the BDI framework provides a very good structure for programming control mechanisms for agents.

To support the concurrent development we use the freely available source code management system CVS [10] and the documentation system doc++ [11].

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.

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] Burkhard, H.-D., Hannebauer, M. and Wendler, J.: Belief-Desire-Intention Deliberation in Artificial Soccer. *AI Magazine* 19(3): 87–93. 1998.
- [3] Gugenberger, P. , Wendler, J., Schröter, K., Burkhard, H. D.: AT Humboldt in RoboCup-98 (Team description). To appear in RoboCup-98 (Springer).
- [4] The sources of AT Humboldt 98 are available from our web pages:
www.ki.informatik.hu-berlin.de/RoboCup/RoboCup98/index_e.html.
- [5] H.D.Burkhard, J.Wendler, H.Myritz, G.Sander, T.Meinert, M.Hannebauer: BDI Design Principles and Cooperative Implementation — A Report on RoboCup Agents. Accepted for Workshop “RoboCup”, IJCAI 99.
- [6] Lenz, M., Bartsch-Spörl, B., Burkhard, H. D., Wess, S.: Case Based Reasoning Technology. From Foundations to Applications. LNAI 1400, Springer 1998.
- [7] Müller-Gugenberger, P. and Wendler, J.: *AT Humboldt 98 — Design, Implementierung und Evaluierung eines Multiagentensystems für den RoboCup-98 mittels einer BDI-Architektur*. Diploma Thesis. Humboldt University Berlin, 1998.
- [8] 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.
- [9] 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*.
- [10] CVS: <http://www.loria.fr/molli/cvs-index.html>
- [11] DOC++: <http://www.zib.de/Visual/software/doc++/index.html>