Mainz Rolling Brains

Daniel Polani and Thomas Uthmann

Institut für Informatik, Johannes Gutenberg-Universität,
D-55099 Mainz, Germany
{polani,uthmann}@informatik.uni-mainz.de
http://www.informatik.uni-mainz.de/PERSONEN/{Polani,Uthmann}.html

1 Introduction

Our agent team is the result of a development which had to take place under tight time limitations. The total development time available was slightly less than three months where over most of the time the team developers could invest no more than a few hours per week. The code was developed from scratch to improve over the design and quality of last year's code. Thus one of the challenges was to keep a smooth development line and to avoid dead ends in the development, as well as to maintain a development environment in which a larger number of developers could work productively.

Our main challenges were twofold: to design an agent architecture which enables robust development of strategically operating agents with a larger developer team; and a lightweight implementation of agents (e.g. running the whole team on a single Ultra 1 Sun machine in Stockholm caused only a quite moderate degradation of the team's performance).

The team Mainz Rolling Brains (MRB) participated in Stockholm for the second time. Like the last year, it reached the fifth position in the total ranking, among that defeating the last year's champion team CMU '98, but succumbing to this year's winner CMU '99.

2 Team Development

Team Leaders: Daniel Polani and Thomas Uthmann
Team Members:

Christian Meyer (graduate, R)	Peter Dauscher (PhD)
Erich Kutschinski (graduate)	Tobias Jung (graduate)
Axel Arnold (graduate)	Sebastian Oehm (graduate, R)
Götz Schwandtner (graduate, R)	Frank Schulz (graduate)
Manuel Gauer (graduate, R)	Achim Liese (graduate)
Birgit Schappel (graduate, R)	Michael Hawlitzki (graduate)
Tobias Hummrich (graduate, R)	Peter Faiß (graduate)
Ralf Schmitt (graduate)	

(Each student's position is marked *graduate* or *PhD*, respectively; *R* indicates attendance at RoboCup '99)

Web page: http://www.informatik.uni-mainz.de/ANGEW/robocup.html

3 World Model

The agents are aware of their own position, of the ball position and the position of other agents, their own speed, the velocity of other agents and objects and of the age (i.e. de facto reliability) of data from objects. This is particularly important for objects not being updated by current data. The world model contains a time window of past and future world states. The past ones are filled in during the run when they are received from the server, the future ones are generated on request of the strategy unit from the most reliable known data. For every object in the game (ball or soccer player) an absolute position representation is used. The player determines his own position using the flags and reconstructs the positions of other objects combining the sensor data giving their relative positions and his own absolute position. Future positions are extrapolated by simulating the known soccer server dynamics for the ball and for the current player. Other players are simulated as not moving. The `libsclient` library is not used in our team.

4 Communication

Our agents only transfer minimal information about status and intention of a player that enables to improve playing quality. The fundamental doctrine of our team using communication is that it should enhance the quality of the play, but the players should in no way depend on it. Disabling communication, our players are still able to act independently, maintaining coherence of the team's play on a purely behavioural level.

Our team's messages are endowed with a checksum and slightly encoded to make sure that only our own team's messages are parsed. Though sending intentionally confusing or forged messages is deemed unfair play in RoboCup, it can still happen that without encoding the agent parser can be trapped on an innocuous message string emitted by the opponent team.

The agents send messages saying whether: they want the ball, they do not want the ball, they want to pass the ball, they are going to a given position, they are low on stamina or they estimate the offside line at a certain field value

An agent without the ball communicates that he wants the ball depending on its estimate of the situation. It can be issued by an agent if a team member is assumed to be in control of the ball. High stamina, far away opponents and being closer to the opponents goal are factors which favor an agent asking for a ball. Also the existence of a broad enough corridor between the player controlling the ball and the agent asking for the ball. The opposite factors (low stamina, close by opponents, lack of a safe corridor between sender and receiver) cause an agent to ask for not receiving the ball.

If a player intends to pass the ball, he communicates this intention, so that another player can issue a request for a ball. No handshake, however, is performed, i.e. the player does not necessarily wait for a reply.

After performing a *catch*, the goalie has the possibility to freely move to a position. By sending the *pass* message the team is informed that the goalie is indeed going to kick the ball away now. Another important situation where an agent emits a *pass* message is when a situation is considered defensive and opponents are close to the player controlling the ball.

When the available stamina is low (our agents stay always above the stamina limit which yields full recovery) *and* an opponent is close to the ball (this is a condition determined by a fuzzy rule), an agent emits the *stamina low* message. For any team player that receives the message this strongly reduces the priority of passing to the player who broadcasted it.

A player who intercepts a ball communicates his estimated interception point to his team by the *going to ball* message. This information is useful first to avoid other team members clustering around the ball and second for giving them the opportunity to optimize their positioning.

A player which has just passed should not immediately request the ball back again; this is a constellation which would be favoured by mainly reactive architectures as ours. Thus, after passing, the priority of requesting the ball back is lowered for a certain time. This, however, does not affect however the respective players behaviour: if there is danger that the ball would be lost, the player will intercede and try to secure the ball, following the doctrine mentioned above.

5 Skills

The most important higher skills available to our agents are *compound kicking*, *intercepting*, *dribbling* and *prepare kick-in*. Compound kicking includes all types of kick combinations which allow a player in reach of the ball to kick it, given a position and velocity, to some other given position and velocity (if possible). As intercepting and dribbling it may include a combination of kick and move commands to realize the desired effect. Our agent design has a strong emphasis on reactive behaviour. The mentioned compound skills, however, require processes that take place during a larger number of time steps. So, one of the most important aspects of our skill design is the realization of *pseudo-multithreads*. On the one side this yields the possibility to carry out compound actions for several time steps in a consistent way. On the other side the agents are able to interrupt a running action to either choose a different one if the situation requires a policy change or to adjust a currently running compound action.

The pseudo-multithread concept works the following way: every time step the agent considers to perform some action according to its strategy. When the agent calls a compound skill method, this calls an elementary action (like e.g. *kick*, *turn*, *dash*). The skill then returns a status value indicating whether: 1. the action was completed by the current call, 2. whether it requires further calls to be completed or 3. whether it is not possible at all to perform this action. In case 1, the skill called in the next time step can be chosen independently from the current step. In case 2, the status value indicates that the skill method has to be called again in the next time steps for the action to be complete. It also gives a reason which can be *wait* (e.g. if player intercepting a ball has not reached its target yet), *collision* which indicates that an intermediate step has been taken to avoid a collision of player and ball or *placing* which indicates that the intermediate step serves to place the ball or agent at a position useful for later kicking.

Due to this mechanism the skills always return control to the agent strategy level. This, in turn, may consider either performing an action thread until it is completed or interrupting it, as considered useful. On a *collision*/*placing* return status, it is usually a disadvantage to interrupt the action thread. In these cases the intermediate states may leave the ball at a position which are only adequate in a predetermined sequence of actions. For the new action sequence this position may be inconvenient.

The compound kick skill in the current version consists of a sophisticated hand-optimized sequence of ball kicks which can not be described in detail here. It attempts to perform a kick that will — from a given ball position and velocity — kick the ball to a given ball velocity in the smallest possible number of kicks. The dribbling uses a variant of the compound kick skill to kick the ball some steps ahead and follow it. It is possible to specify the average number of time steps that the player is dashing before he emits a kick. The higher this number is, the less contact the player has with the ball. It is also possible to specify a preferred direction with respect to the player where the ball is supposed to be kept.

6 Strategy

The strategy is behaviour oriented, i.e. communication is useful but not necessary for the capability of teamplay. Fuzzy predicates characterize a given situation. The predicates are then combined with fuzzy rules resulting in the respective action choice. If an agent controls the ball, it estimates the situation, i.e. the pressure by opponents, the support by team members and its own stamina state. Depending on these parameters, it will try to move the ball ahead in the field (preferably at the wings) and to kick the ball towards the centre near the opponent goal. Opponent pressure or low stamina will increase its priority to pass the ball.

If the players do not have the ball, their action is determined in part by their role (giving a preference position), in part by the current situation. A player whose *responsibility* (a fuzzy predicate) for the ball is high will dash to the ball. If his responsibility is low he will try to acquire a good position (e.g. a position giving a good opportunity for passes by a partner). If opponent players are close to the ball, players will feel responsible more easily, so that more agents will try to intercept the ball.

7 Special Team Features

In the current stage no parts developed via machine learning methods are used. Neither are opponent models used. After completion of the server communication and world model modules several independent lines of development for the strategy were taken while the agent skills module was optimized in parallel. About three weeks before the tournament a decision was taken in favour of one of the strategy development lines to be used in the official agent team. Strategical ideas from the other lines of development were then incorporated in the main line.

8 Future Work

The agent team presented here has been developed in such a way to consider the needs of a large developer group; it includes clear and well-documented module interfaces, in particular of world model and skills. This makes it amenable for further development and a participation at RoboCup 2000 is envisaged. Relevant aspects of our future work include the improvement of ball control and the positioning tactics via machine learning techniques.