Team Description
Mainz Rolling Brains 2004 - 3D

Axel Arnold, Felix Flentge, Christoph Schneider

Department of Computer Science
Johannes Gutenberg-University Mainz
D-55099 Mainz, Germany

1 Introduction

Since we had little time getting familiar with the rcssserver3d and the rules are still not fixed definitely, the focus of the following paper is on our overall agent design. We are going to describe how we want to carry over the design of our 2D agent and which problems will be tackled in near future.

2 Agent Design

There are several requirements for the agent design. As we are going to work with about ten students on our agent, it should be easy to split the work in small tasks and to coordinate all activities. Also `rcssserver3d` will change over the years, e.g. we will have more complex agent shapes, different effectors and perceptors. So there is the need for a design which easily readjust with the certainly growing demands of the server. Because of the good experiences with our 2D agent design during the last years, we will try to carry over the design principles of the 2D agent(also see the team description of the 2D team).

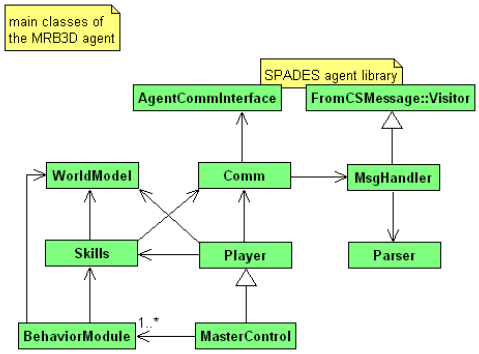

Fig. 1. Agent Design Overview

2.1 Communication

As the basic simulation engine is based on SPADES, we use the SPADES agent library from P. Riley [Riley03] to handle the communication with the server. For our first sample agent we use the same communication scheme as for our 2D agent, i.e. the agent waits for a message from the server and updates its world model according to it. Then the player selects an action based on the new information, sends it to the server, and starts waiting for a new message from the simulator. This could be changed to a multi threaded model to decide upon arrival of a new message whether to finish the current decision process or to start a new one based on the new information.

2.2 World Model

The world model provides the agent with all information he needs to know, e.g. positions or velocities of all objects on the field. It is also responsible for the quality of these informations, i.e. to take the various sources of errors into account, e.g. the visual sensor's calibration error. For self-localization we plan to transfer the algorithm used in our 2D agent [MRB01a] based on intersecting polygons (which in 3D will become polyeders).

2.3 Skills

The *Skills* are an abstraction to low level server commands and should provide the decision layer with high-level commands, like 'go to a certain position' or 'kick the ball in a certain direction with a certain speed'. While in the old 2D soccer system physics were discrete and therefore a discrete optimization was necessary, in the 3D server system physics are practically continuous. We plan to use dynamic feedback to provide us with sufficiently precise velocity information (as well of the relative ball movement as of our own speed). Once this information is obtained, we will use a hard coded optimization as was used before in the 2D agent, so 3D physics will only change the constraints. For example, the ball can no longer go through a player. We expect this to decrease the complexity of the optimization used in ball placing, kicking, and dribbling. The intercept skill (either of the ball or a position) will be implemented in a kind of greedy fashion, i.e. drive in the direction of the target and slow down soon enough.

2.4 Decision Layer: Behavior Modules and *MasterControl*

The decision layer will be based on our well-proven modular decision layer concept ([MRB00a], [MRB01b], see also our 2D team description paper). We expect to have these five modules:

- The *BallHandling* module for intercepting the ball and dribblings.
- The *Pass* module for deciding to which teammate the ball should be passed to and for executing passes.

– The *GoalShot* module for shooting a lot of goals.
– The *Positioning* module for the positioning of the players on the field depending on the ball possession (defense / offense).
– The *StandardSituation* module for handling all game states except 'PlayOn'.

Of course, the final shape of these modules will depend on the rules for this year's competition, e.g. positioning highly depends on the existence of an offside rule. But all modules will have an `evaluate` procedure which returns a discrete grade as a measure for the adequateness and the probability of success for their respective task in the current situation. And all modules will contain an `act` procedure for executing their tasks using the *Skills*.

It is an open question how timing issues will be handled in this model, as the new server can take the thinking time of an agent into account. This should be handled as far as possible by the individual modules themselves. Maybe we will need some way for a module to tell the *MasterControl* that its `act` method has to be called without calling any other modules to finish an action which could otherwise be disturbed due to time constraints. But, in general, it may be better to let each module evaluate the situation every cycle to allow the interruption of currently executed actions. For example, if the pass module is preparing a pass, the goalshot module may discover an opportunity for a perfect goalshot and therfore should take over the control. If thinking time turns out to be critical, one way would be to have different evaluation procedures in the modules, e.g. one for quick evaluations and one for full evaluations. But all these issues depend on the final rules and parameter settings for this year's competition.

3 Conclusion

The paper has shown how we want to carry over the design principles of our 2D agent to the new server. We also gave some hints on how we want to solve the new problems arising in the 3D world. We think a robust but flexible agent design as presented is the best basis to tackle these problems.

References

[Riley03] Riley, P., Riley, G.: SPADES — A Distributed Agent Simulation Environment with Software-in-the-Loop Execution. In: Winter Simulation Conference Proceedings (2003) 817–825

[MRB01a] Arnold, A., Flentge, F., Schneider, Ch., Schwandtner, G., Uthmann, Th., Wache, M.: Team Description. Mainz Rolling Brains 2001. In: RoboCup-01. Robot Soccer World Cup V, Springer, Berlin Heidelberg New York (2002)

[MRB01b] Flentge, F., Meyer, C., Schappel, B., Uthmann, Th.: Enhancing the Adaptive Abilities. Mainz Rolling Brains 2001. `http://www.rollingbrains.de/MRB01_2.ps`

[MRB00a] Schappel, B., Schulz, F.: Mainz Rolling Brains 2000. In: Stone, P., Balch, T., Kraetzschmar, G. (ed.): RoboCup 2000: Robot Soccer. World Cup IV. Lecture Notes in Computer Science, Vol. 2019. Springer, Berlin Heidelberg New York (2001)