RoboCup-99 Team Descriptions
Simulation League, Team *UvA-Team*, pages 147–151
http://www.ep.liu.se/ea/cis/1999/007/33/

Team description of the UvA-Team

UvA-Team

Emiel Corten, Frans Groen

EC, FG: Intelligent Autonomous Systems, WINS, University of Amsterdam

Abstract. *This paper describes the architecture and progress in the development of the UvA soccer simulation team. The UvA-team uses a three layer model to separate simulator details, basic soccer skills and the overall control from each other. In the previous year we focused on getting the system up-and-running. The current effort is in the use of learning mechanisms for optimizing the parameters of the soccer skills like dribbling and shielding. We are also applying the capabilities of the coach to characterize an opponent on the basis of logfiles to select suited strategies against that opponent as well as to adapt parameters of the control and sensor systems during the play.*

1 Introduction

The UvA-team, formerly known as the Windmill Wanderers, made a good performance at the Paris'98 RoboCup simulation competition, and reached the 3th place. We believe that the basis of our success was the development of some powerful soccer skills like shooting hard, dribbling and shielding the ball from opponents as well as a basic but effective zone strategy.

In this paper we will give a short description of the three layers used. A more detailed description can be found in the team description for Paris'98 [**?**]. The parametrization of the skill behaviours, feature extractors and the control section proved to be good enough to make a working soccer team.

Our current research focusses on methods to learn the optimal values for those parameters. In the case of the skill behaviours we have developed a system that uses a generic algorithm to find the optimal parameters for those behaviours while performing short training sessions. Also, we are developing a coach-based opponent classification system. Our aim is to use this system in various ways. First, to be able to automatically assign the optimal parameter values to feature extractors like 'how free most a teammate be to be able to receive the ball' and 'how open must the goal be to have some chance on scoring'. Second, to use the classification based on logfiles of the opponent to select for instance one out of several prepared zone setups. Third, to dynamically adjust parameters during the match based on observed behaviour of the opponent.

This paper is organized as follows. In section 2 the three layers are described. In section 3 we will explain the zone strategy. In section 4 we will describe the learning systems used for our team. Finally we give some conclusions.

2 Architecture

2.1 Basic layer

The BasicPlayer layer hides the soccer simulation server as much as possible from the other layers. It provides access to all functionality offered by the soccer server. Besides subsystems to send the actuator commands to the server and to receive and parse the sensor input from the server we added a visual memory and an internal state subsystem. With the subsystems chosen we expect that the BasicPlayer can be used as a convenient starting point for working with the soccer simulation without constraining the user in his or hers intentions.

2.2 Skills layer

The SkilledPlayer layer uses the functionality offered by the BasicPlayer to implement the following functions: essential tasks, elementary tasks and feature extractors.

Essential tasks Using our soccer instincts we distinguished a limited number of purpose-directed tasks for the game of soccer: Intercept, Shoot, Pass, Shield, Dribble, Defend, Determin next (free) target position.

Elementary tasks An elementary task does its work in a limited number of basic actions (dash, turn etc). It is a task that usually has no intrinsic purpose. Most of the time it is activated while performing an essential task. Elementary task can use (extracted) features to guide their actions. Examples of such tasks are 'search the ball', 'move the ball around within the kicking area of the client' and 'perform multiple kicks to shoot or pass'.

Feature extractors A feature is a derived piece of information that it is important when deciding what to do next. Examples are 'team mate free', 'goal free' or 'closest to ball'. It can also be information, like 'there is an opponent to the left of me', which is used to decide how to perform a certain action. All these features are calculated using the current information available in the Visual Memory and Internal State.

2.3 Control layer

The task of the control layer is to select the optimal action from the skills layer given a certain state of the game (including previous states). Our current control layer consists of a if-then-else tree that implements a decision tree.

The control layer makes use of parameters in two ways. The boundaries for the conditions are parameterized. But also the values tested in the decisions

are implicitly parameterized because the feature extractors from the skills layer are.

The main loop of the control layer looks like:

```
1. (start of cycle) If new visual input needed, wait for it
2. Calculate decision-related features.
3. Decide which essential task to trigger, and trigger it
4. Next cycle
```

Because all actions that can be triggered ars such that they take only a limited number of cycles per activation the control layer can quickly react to changing situations.

3 Zone strategy

An important aspect in playing soccer is to decide what to do when one does not have control over the ball and decided not to go to the ball. We have developed and implemented a simple but effective zone strategy that enables the players to keep a good distribution over the field. At the same time the individual actions taken by the players are such that the positions of defenders, midfielders and attackers follow the location of the ball position in a natural way.

3.1 Global positioning

Each player has its own subdivision of the field into up to 10 x 10 subfields and has received a rectangular zone-of-most-interest specification. To each subfield are assigned two functions from a set of possible functions that respectively give the best coordinates in the length (x) and width (y) of the field to go using the current location of the ball as main input. Examples of functions for the 'length' direction are: as far to the back/front of your zone as possible, stay at certain offset from the x-coordinate of the ball. Examples for the 'width' coordinate functions are: At the y-coordinate of the ball, between own/other goal and ball, to the side of your zone closest/furthest from the ball.

The functions of the subfield in which the ball is currently found are used to calculate the current global target position.

3.2 Movements around global position

The zone directions given by the system described above only gives a ball-related indication where the player could/should go to. When the player is at or around the indicated position the actual state of the game, positions of other players, must be used to choose the actions to undertake to optimize the possibility to receive the ball (midfielder and attacker), to prevend an opponent to get the ball (defender, midfielder), etc.

We are developing a potential field-like attraction-repulsion system to improve the 'local' positioning of the players. The attractive and repulsive forces are supplied by other players and indirectly by feature extractors that indicate favourable locations.

4 Applications for learning mechanisms

4.1 Optimizing parameterized skills

We have developed a number of soccer skills like shooting, dribbling and shielding of the ball. The number of parameters used to control these skills are quit large. Although the hand-tuned values of the parameters we used in Paris'98 resulted in satisfatory behaviour we are sure that learning the optimal values in an automated way is a much better approach.

We have implemented a system that uses a generic algorithm to find the optimal parameters values for a parameterized task. The system consists of three parts: The coach application, the player application for which some skill has to be improved and the generic algorithm application.

It is important to note that the player itself needs minimal awareness of the fact that it is being trained. The learning system only requires from the player that it can change parameters using some kind of string command interface. The parameter commands themselves can for instance be transmitted using the 'say' messages available in the soccer simulation. To optimise a task one must further define a training in which the task is used and an evaluation function that judges the performance of training sessions. The coach is used to set-up the training (positioning the players, starting and stopping) and to evaluate a training session.

4.1.1 First results

We have tested this system by optimizing the shooting skill. This is a behaviour with an easy evaluation function. And because it has relatively little parameters a exhoustive search was feasible as well.

Then we proceeded to optimize the dribble behaviour both with and without opponents. Both resulted in faster and safer dribbling behaviour as compared to the Paris'98 behaviour.

4.2 Classification of opponents and on-line adaptation of parameters

We are going to use the coach to observe the game and/or logfiles and let it autonomously determin statistics that describe how one or both teams play. These statistics may be descriptions of player capabilities like how fast do they ran, how good is the intercept. They may also be more qualitive statistics describing the play itself. These may depend on the positions and the change of positions of the ball and of one or more players. Also more subjective (derived) statistics can be used like for instance ball possession.

This scouting/observation system is going to be used for:

Classification Try to classify an opponent to select the best strategy of the own team against that opponent using logfiles that are available of the opponent. Strategies that are likely to be optimized are: setup of the zone and parameter values that control the balance between passing, dribbling and shooting.

Adaptation of parameters The coach observes the game and extracts statistics about certain capabilities of the opponent. It can use these statistics to adapt parameters that are used by feature extractors like 'team mate free' or 'goal free'. The results presented by developers of the ISAAC system [**?**] show that this approach can improve the performance of a team.

Evaluation of training sessions In the future we would like to optimize the team as a whole using for instance GA or reinforcement learning. To be able to do this we are matching one but probably more of the statistics above with statistics as given by a human observer, for instance by telling which team he/she thinks is better. These statistics can then be used for (input of) an evaluation function.

5 Conclusions

We have discussed the 3-layer architecture of the UvA simulation soccer team and the learning mechanism used to optimize parameterized skills of individual players. Using a learning mechanism and the need of an observer function to evaluate the succes of a specific implementation and choice of parameters during a training has a number of advantages. It is easy to retrain the parameters after changes in enviromental settings (like player size) or in the implementation of the skill behaviour or elementary tasks used by the skill. Furthermore, a well chosen evaluation function can be used to compare alternative approaches to accomplish the same training target.

To extend the use of a learning mechanism to optimize not only to skills in isolation but to 'playing soccer' as well we need a system that can evaluate at 'game level'. The system we discussed in this paper must provide this evaluation function. We expect that a subset of the statistics that this system will derive from a game can also be used to dynamically change parameters of the players in order to adapt to an opponent during a game.

Until now the team behaviour is mainly an implicit result of the zone-strategy used by the individual players. In future work we will concentrate on explicit multi-agent behaviour as well.