Brainstormers 2D — Team Description 2008

M. Riedmiller[1], T. Gabel[1], F. Trost[2], T. Schwegmann[1]

[1] Neuroinformatics Group
Institute of Mathematics and Computer Science
Institute of Cognitive Science
Universität Osnabrück, 49069 Osnabrück, Germany
{martin.riedmiller|thomas.gabel|tobschwe}@uos.de
[2] Gymnasium Carolinum, 49074 Osnabrčk, Germany
floriantrost@gmx.net

Abstract. The main focus of the Brainstormers' effort in the RoboCup soccer simulation 2D domain is to develop and to apply machine learning techniques in complex domains. In particular, we are interested in applying reinforcement learning methods, where the training signal is only given in terms of success or failure. Our final goal is a learning system, where we only plug in "win the match" – and our agents learn to generate the appropriate behavior. Unfortunately, even from very optimistic complexity estimations it becomes obvious, that in the soccer simulation domain, both conventional solution methods and also advanced today's reinforcement learning techniques come to their limit – there are more than $(108 \times 50)^{23}$ different states and more than $(1000)^{300}$ different policies per agent per half time. This paper outlines the architecture of the Brainstormers team, focuses on the use of reinforcement learning to learn various elements of our agents' behavior, and highlights other advanced artificial intelligence methods we are employing.

1 Introduction

The Brainstormers project was established in 1998, starting off with a 2D team. Ever since we have been participating in RoboCup's annual soccer simulation tournaments. Over the years, the Brainstormers Tribots (competing in RoboCup's MidSize league since 2002), the Brainstormers 3D (soccer 3D simulation, 2004–2006), as well as the Brainstormers Twobots (Humanoid League, since 2008) expanded the Brainstormers team. Our work has been accompanied by the achievement of several successes such as multiple World Vice Champion titles and the World Champion titles at RoboCup 2005 in Osaka (2D), RoboCup 2006 in Bremen (MidSize), and RoboCup 2007 in Atlanta (2D + MidSize).

The team description paper at hand focuses on the Brainstormers 2D, our team competing in soccer simulation's 2D league. The underlying and encouraging research goal of the Brainstormers has always been to exploit AI and machine learning techniques wherever possible. Particularly, the successful employment of reinforcement learning (RL) methods for diverse elements of the Brainstormers' decision making modules has been and is our main focus as shall be detailed in the subsequent sections.

1.1 Design Principles

The Brainstormers 2D rely on the following basic principles:

- There are two main modules: the world module and the decision making module.
- Input to the decision module is the approximate, complete world state as provided by the soccer simulation environment.
- The soccer environment is modelled as a Markovian Decision Process (MDP).
- Decision making is organized in complex and less complex behaviors.
- A large part of the behaviors is learned by reinforcement learning methods.
- Modern AI methods are applied wherever possible and useful (e.g. particle filters are used for improved self localization).

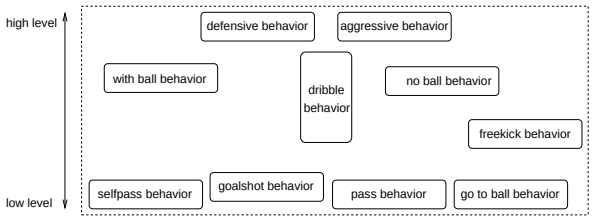

Fig. 1. The Behavior Architecture

1.2 The Brainstormers Agent

The decision making process the Brainstormers Agent is based upon is inspired by behavior-based robot architectures. A set of more or less complex behaviors realize the agents' decision making as sketched in Figure 1. To a certain degree this architecture can be characterized as hierarchical, differing from more complex behaviors, such as "no ball behavior", to very basic, skill-like ones, e.g. "pass behavior". Nevertheless, there is no strict hierarchical sub-divisioning. Consequently, it is also possible for a low-level behavior to call a more abstract one. For instance, the behavior responsible for intercepting the ball may, under certain circumstances, decide that it is better to not intercept the ball, but to focus on more defensive tasks and, in so doing, call the "defensive behavior" delegating responsibility for action choice to it. With the intention to make a contribution to the entire soccer simulation community – in particular, to new teams for which, as is known, it is difficult to overcome basic problems, such as developing a reliable world model or basic skills – our team's source code has been made publicly available and can be retrieved from our team web site[3].

[3] http://www.brainstormers.uos.de

2 Recent Research Efforts and Developments

After having outlined the basics on the Brainstormers' competition team, we now want to give an overview on recent developments and on one specific reinforcement learning approach that significantly enhanced our team's capabilities. A more comprehensive review of our efforts on utilizing neural reinforcement learning methods in the scope of the Brainstormers 2D can be found in [1].

2.1 Duel Behavior: The `NeuroHassle` Approach

"Aggressive playing" is a buzz-word frequently used in today's media coverage of human soccer-playing. By aggressiveness it is usually referred to a player's willingness to interfere the opponent team's game build-up early and, in particular, to quickly and efficiently hassle and attack opponent ball leading players.

The Brainstormers' former approach for letting players duel with opponent ball leaders for the ball was a rather naive one: The player next to the opponent ball leading player simply moved towards the ball leader and towards the ball, respectively, in order to try to bring the ball into his kickable area. Needless to say that such a straightforward strategy is not difficult to overcome. Consequently, our agents failed in conquering the ball frequently – in particular when playing against teams with highly developed dribble behaviors.

A general strategy to hassle an opponent with the goal of conquering the ball is difficult to implement because

- the task itself is far beyond trivial and its degree of difficulty heavily depends on the respective opponent,
- there is a high danger of creating an over-specialized behavior that works well against some teams, but performs poorly against others, and
- duels between players (one without and the other with the ball in his possession) are of high importance for the team as a whole since they may bring about ball possession, but also bear some risk, if, for example, a defending player looses his duel, is overrun by the dribbling player, and thus opens a scoring opportunity for the opponent team.

To handle these challenges holistically, we decided to employ a neural reinforcement learning approach that allows our players to train the hassling of opponent ball leading players. This approach is described in detail in [2]. Here, we provide a short summary of our *NeuroHassle* case study only.

2.1.1 Problem Modelling

The state space for the problem at hand is continuous and high-dimensional; we restricted ourselves to 9 state dimensions: the distance d between our player and the opponent ball leading player, the velocity (v_x and v_y component) of our player, the absolute value v_{opp} of the opponent's velocity, the position (b_x and b_y component) of the ball, our player's body angle α relative to the opponent's position, the opponent player's body angle β relative to his direction towards

our goal, and, finally, the value of the strategic angle $\gamma = \angle GOM$ with G as position of our goal, O as position of the opponent, and M as the position of our player

The high dimensionality of the problem space requires the use of value function approximation mechanisms if we aim at applying value function-based RL. To this end, we rely on multilayer perceptron neural network. The learning agent is allowed to use $dash(x)$ and $turn(y)$ commands where the domains of bots commands' parameters are discretized such that in total 76 actions are available to the agent at each time step.

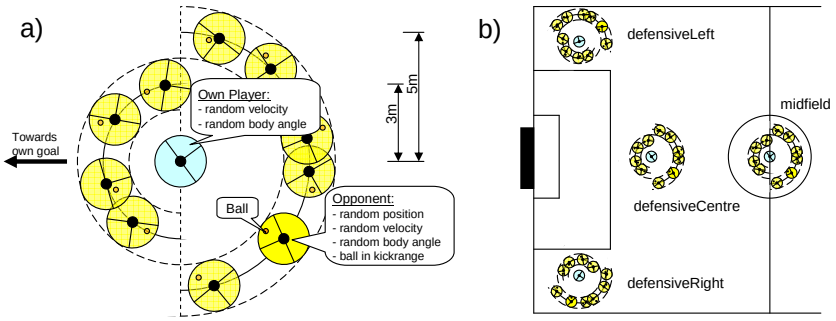

Fig. 2. General Training Scenario for Learning to Hassle Opponent Ball Leading Players

We designed a specialized set of *training situations* S for the learning agent ($|S| = 5000$) which is visualized in Figure 2a. The intention behind that design of starting situations is that the intended hassle behavior shall be primarily applied in situations where our player is closer to our goal or where the opponent ball leader has only a small head start.

Moreover, we defined four *training regions* on the playing field as sketched in Figure 2b. The midfield training region is situated at the center of the field, the central defensive region is halfway on the way towards our goal. Finally, there are a left and a right defensive region that are placed near the corners of the field with a distance of 25 meters to our goal. The idea behind this definition of different training and testing places is that dribbling players are very likely to behave differently depending on where they are positioned on the field. As a consequence, a duel for the ball may proceed very differently depending on the current position on the field.

A highly important issue concerns the question whether a single hassling episode, i.e. a single duel for the ball, was successful or not. After a careful analysis of the learning setting and of the way opponent ball leading players may behave during training, we found that five main outcomes of a training episode must be distinguished: (a) erroneous episodes (episodes that fail, e.g. due to the dribbling player may losing the ball), (b) success episodes (ball has been brought into the learning player's kickable area or tackle becomes promising),

(c) opponent gets into panic (ball leading opponent simply kicks the ball away, usually forwards; episode may be regarded as a draw), (d) failure episodes (none of the other cases has occurred, i.e. ball leader has kept the ball in its kick range, or has even overrun the learning agent and escaped more than $7m$ from him), (e) time out (we allow a maximal episode duration of 35 time steps).

2.1.2 The Learning Algorithm

Learning to hassle, we update the value function's estimates according to the temporal difference learning $TD(1)$ update rule [3], where the new estimate for $V(s_k)$ is calculated as $V(s_k) := (1-\alpha) \cdot V(s_k) + \alpha \cdot ret(s_k)$ with $ret(s_k) = \sum_{j=k}^{N} r(s_k, \pi(s_k))$ indicating the summed rewards following state s_k and α as a learning rate. Each time step incurs small negative rewards, a success goal state a large positive one, and the final state of a failure episode a large negative one.

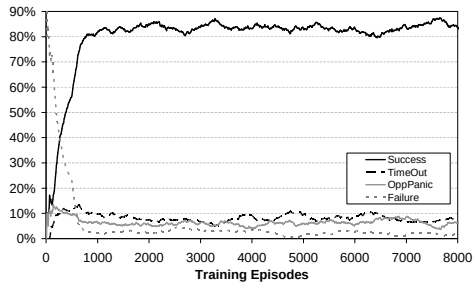

Fig. 3. Exemplary Learning Curve for Learning to Hassle (opponent during training: Wright Eagle, binary from RoboCup 2006)

To approximate the value function, we employ multi-layer perceptron neural networks with one hidden layer (9:18:1-topology). We perform neural network training in batch mode: Repeatedly a number of training episodes is simulated and in so doing a set of representative states $\tilde{S} \subset S$ is incrementally built up where for each $s \in \tilde{S}$ we have an estimated value $V(s)$ calculated as mentioned above. Let the state value function approximation provided by the net be denoted as $\tilde{V}(s, w)$ where w corresponds to a vector of tunable parameters, i.e. the net's connection weights. Then, the actual training means determining w by solving the least squares optimization problem $min_w \sum_{s \in \tilde{S}} (\tilde{V}(s, w) - V(s))^2$. For the minimization we rely on the back-propagation variant *RPROP* [4].

2.1.3 Evaluating the NeuroHassling Performance

Figure 3 shows the learning curves for an exemplary hassle learning experiment. Here, the neural network representing the value function from which a greedy policy is induced has been trained against the binary of WrightEagle (2006) for midfield training situations. Apparently, the initially clueless learning agent quickly improves its behavior and finally succeeds in successfully conquering

the ball in more than 80% of all attempts. In particular, the number of failure episodes is extremely reduced (to less than 5%) which highlights the effectiveness of this learning approach.

In [2] we report the performance of the acquired policy against various test opponents and for various test scenarios. Employing the learned *NeuroHassle* policy was one of our crucial moves for winning the World Championships tournament RoboCup 2007 in Atlanta.

2.2 Flash Animations of RoboCup 2D Soccer Simulations

Watching logfiles of recorded 2d soccer simulation matches using the standard `rcssmonitor` and `logplayer`, sometimes leaves the unacquainted viewer puzzled. Hence, providing animations that, at least in part, resemble real soccer is a must. There has been already some work into that direction. For example, [5] developed a tool called `robocup2flash`, that allowed for an easy way to watch a simulated game properly. However, the animations produced looked not very appealing and resembled the `rcssmonitor` look a lot.

Therefore, it was our goal to provide a better looking and more realistic animation of a simulated soccer game. The result of these effort is a tool called `rcg2swf` [6][4] that is capable of creating a handsome visualization of a RoboCup Soccer 2D Simulation game. Using `rcg2swf` to create an animation, for example, the ball now looks like a real ball and not just like a white circle, the players look more like a human and are not colored circles. The use of flash animations makes it possible that a Robocup Soccer 2D simulation can now be seen as a football game not only interesting for RoboCup related persons, but also for a broader audience. Like in a real soccer stadium we also have a display showing the time, team names, standings, and the actual state of the game. Furthermore, there are places for advertisements where the sponsors of the teams can be represented. Figure 4 shows a screenshot of a `rcg2swf` animation and contrasts it with the look of `rcssmonitor` and `robocup2flash`.

The base for the tool are the logfiles (`rcg` files) the RoboCup Soccer Server creates. The data is read with a logfile reader and, with the help of the Macromedia Flash API, the animation can be created. Unlike `robocup2flash` there are many different classes who represent the several objects of the animation, this makes it a lot easier to make changes or improvements to `rcg2swf`.

There is also a possibility to configure the look of the animation, you can change colors of the teams, set other advertisements, or change the size of the animation arbitrarily. You can also define different scenes, so instead of watching the whole game you can watch only some scenes of the game with an introducing text – which is very useful for creating animations that show the best scenes of a match. As another highlight, slow-motion is added to the animation: After a goal has been scored a replay of this scene is shown. Further features of `rcg2swf` include an automatic side change of the teams (as in real soccer) after the first half time and a number of buttons to control the flow of the animation.

[4] http://www.ni.uos.de/index.php?id=1024

Fig. 4. Different approaches to visualize soccer simulation 2D matches: standard soccer monitor (`rcssmonitor`, top left), animation by [5] (top right), and `rcg2swf` (bottom).

3 Past, Present, Future

This year, the Brainstormers team celebrates its 10th birthday. Over the years, numerous people have contributed to the team's steady further-development. Also, from year to year the scientific focus has shifted to a smaller or larger extent. Table 1 provides a comprehensive overview on the Brainstormers' behaviors learned by neural reinforcement learning methods. The upper part of the table shows the individual skills, the lower part shows the multi-agent skills. Filled circles ('•') denote the years, where the learned skill was actually used in the Brainstormers competition team at the World Championships of RoboCup. Empty circles ('○') denote the years, where a certain skill was developed or improved, but not used in the competition team[5]. The state space dimensions and action space cardinalities of the problems show that the tasks to be learned are far beyond trivial. The final row shows the ranking that our team achieved at the respective annual RoboCup World Championships tournament.

During the past four years, our team could greatly benefit from facing a nearly stable simulation environment. This allowed us to concurrently (a) redesign vast parts of our team play and (b) enhance several of the machine learning ap-

[5] Demo videos of the learned behaviors and the learning process can be found at our website: **www.ni.uos.de/brainstormers**.

Table 1. Overview of behaviors that were learned by neural RL methods.

	dim(S)	card(A)	2000	2001	2002	2003	2004	2005	2006	2007
NeuroKick	5	1204	•	•	•	•	•	•	•	•
NeuroIntercept	6	76	•	•	•	•		○	○	
NeuroGo2Pos	6	76	•	•	•	•	•			
NeuroDribble	11	282	○						○	
NeuroHoldBall	8	360	•	•						
NeuroHassle	9	76								•
NeuroAttack2vs2	14	13	○							
NeuroPos7vs8	34	10		•						
NeuroAttack3vs4	18	14			○					
NeuroAttack7vs8	34	18			•	•	•		•	•
NeuroPenalty1vs1	8	11				•	•	•	•	•
NeuroScore	18	14					•			
Rank at RoboCup			2	2	3	3	2	1	2	1

proaches we employ in such a manner that the resulting behaviors are highly competitive. Currently, a considerable number of changes is being introduced to the simulation environment (Soccer Server Ver.12). As a consequence, our main focus in 2008 is to adapt our team and coach to the changes introduced.

4 Summary

In this team description paper we have outlined the characteristics of the Brainstormers team participating in RoboCup's 2D Soccer Simulation League. We have stressed that our main research focus lies on the development of reinforcement learning techniques and their integration into our team.

References

1. Riedmiller, M., Gabel, T.: On Experiences in a Complex and Competitive Gaming Domain: Reinforcement Learning Meets RoboCup. In: Proceedings of the 3rd IEEE Symposium on Computational Intelligence and Games (CIG 2007), Honolulu, USA, IEEE Press (2007) 68–75
2. Gabel, T., Riedmiller, M., Trost, F.: A Case Study on Improving Defense Behavior in Soccer Simulation 2D: The NeuroHassle Approach. In: RoboCup 2008: Robot Soccer World Cup XII, LNCS, Berlin, Springer Verlag (2008, to appear)
3. Sutton, R.S.: Learning to Predict by the Methods of Temporal Differences. Machine Learning **3** (1988) 9–44
4. Riedmiller, M., Braun, H.: A Direct Adaptive Method for Faster Backpropagation Learning: The RPROP Algorithm. In Ruspini, H., ed.: Proceedings of the IEEE International Conference on Neural Networks (ICNN), San Francisco (1993) 586–591
5. Giermann, T.: Robocup2flash, http://sourceforge.net/projects/robolog/. (2003)
6. Schwegmann, T.: Flashanimationen von RoboCup Soccer 2D Simulationen. Diploma thesis, University of Osnabrueck (2007)