A Case Study on Improving Defense Behavior in Soccer Simulation 2D: The NeuroHassle Approach

Thomas Gabel, Martin Riedmiller, and Florian Trost

Neuroinformatics Group
Institute of Mathematics and Computer Science
Institute of Cognitive Science
University of Osnabrück, 49069 Osnabrück, Germany
{thomas.gabel,martin.riedmiller}@uos.de, floriantrost@gmx.net

Abstract. While a lot of papers on RoboCup's robotic 2D soccer simulation have focused on the players' offensive behavior, there are only a few papers that specifically address a team's defense strategy. In this paper, we consider a defense scenario of crucial importance: We focus on situations where one of our players must interfere and disturb an opponent ball leading player in order to scotch the opponent team's attack at an early stage and, even better, to eventually conquer the ball initiating a counter attack. We employ a reinforcement learning methodology that enables our players to autonomously acquire such an aggressive duel behavior, and we have embedded it into our soccer simulation team's defensive strategy. Employing the learned *NeuroHassle* policy in our competition team, we were able to clearly improve the capabilities of our defense and, thus, to increase the performance of our team as a whole.

1 Introduction

Over the years, RoboCup simulated soccer has turned out to be a highly attractive domain for applying machine learning and artificial intelligence approaches [1]. Beyond the scientific focus and the goal of beating the human soccer World Champion team in 2050, the annual RoboCup competitions also represent a highly competitive event. Consequently, if a team participating in a RoboCup tournament aims not just at exploring learning approaches in this application field, but also at using player behaviors, that were obtained with the help of some learning approach, during competitions, then there is a natural desire that also the AI-based player capabilities represent optimal or at least near-optimal behavior policies, so that their employment does not worsen the overall quality of playing.

This, of course, also holds true for a team's defense. Research in soccer simulation, however, has to the biggest part focused on offensive playing. To this end, mainly the tasks of passing and scoring [2,3] were addressed, as well as the

L. Iocchi et al. (Eds.): RoboCup 2008, LNAI 5399, pp. 61–72, 2009.

positioning of players not in ball possession [4,5]. Moreover, there were also studies that considered offensive playing in a holistic manner [6,7]. By contrast, the challenges of using learning approaches for a team's defensive capabilities have almost been neglected. A notable exception in this regard represents the work on positioning defending players [8]. Generally, defensive playing strategies comprise two core sub-tasks: On the one hand, there is the task of positioning players in free space so as to intercept opponent passes or to cover or mark opponent players. On the other hand, there must be a player that attacks the opponent ball leading player, interferes him, hassles him, and aims at bringing the ball under his control. While the paper mentioned before [8] focuses on the former task of positioning, the work at hand presents a case study concerning the latter: Using a reinforcement learning (RL) methodology, we aim at acquiring an aggressive player behavior which makes the learning agent interfere and hassle a ball leading opponent player effectively. Because the RL approach employed makes use of neural net-based value function approximation, we call the resulting behavior *NeuroHassle*, subsequently. The policy we obtained clearly surpasses any hand-coded approach for this task and improved our team's defensive strength significantly. With respect to the remarks made at the beginning of this section, a noteworthy property of the acquired hassling policy is its high degree of competitiveness which allowed us to integrate NeuroHassle into our competition team *Brainstormers* and to employ it successfully at RoboCup 2007 in Atlanta.

Section 2 provides some foundations on RL and on the soccer simulation environment. In Section 3, we introduce in detail our learning methodology and algorithm, whereas Section 4 evaluates the learning results empirically and discusses the integration of the acquired hassling policy with our competition team.

2 Basics

One reason for the attractiveness of reinforcement learning (RL) [9] is its applicability to unknown environments with unidentified dynamics, where an agent acquires its behavior by repeated interaction with the environment on the basis of success and failure signals. This situation is, also, given when an initially clueless soccer-playing agent is faced with the task of conquering the ball from an opponent ball leading player (of an adversary team regarding whose dribbling capabilities nothing is known). Accordingly, the usage of RL to tackle this learning problem is very promising. A more comprehensive review of our efforts on utilizing neural reinforcement learning methods in the scope of the RoboCup soccer simulation 2D League can be found in [6,10].

In each time step an RL agent observes the environmental state and makes a decision for a specific action, which may incur an immediate reward generated by the environment and, furthermore, transfers the agent to some successor state. The agent's goal is not to maximize its immediate reward, but its long-term, expected reward. To do so it must learn a decision policy π that is used to determine the best action in any state. Such a policy is a function that maps the current state $s \in S$ to an action a from a set of viable actions A and the

goal is to learn the mapping $\pi : S \rightarrow A$ only on the basis of the rewards the agent gets from its environment. By repeatedly performing actions and observing resulting rewards, the agent tries to improve and fine-tune its policy. The respective reinforcement learning algorithm used specifies how experience from past interaction is used to adapt the policy. Assuming that a sufficient amount of states has been observed and rewards have been received, the optimal policy may have been found and an agent following that policy will behave perfectly in the particular environment.

Reinforcement learning problems are usually formalized using Markov Decision Processes (MDPs) [11], where an MDP is a 4-tuple $[S, A, r, p]$ with S as the set of states, A the set of actions the agent can perform, and $r : S \times A \rightarrow \mathbb{R}$ a function of immediate rewards $r(s, a)$ (also called costs of actions) that arise when taking action a in state s. Function $p : S \times A \times S \rightarrow [0, 1]$ depicts a probability distribution $p(s, a, s')$ that tells how likely it is to end up in state s' when performing action a in state s. Trying to act optimally, the agent needs a facility to differentiate between the desirability of possible successor states, in order to decide on a good action. A common way to rank states is by computing a state value function $V^{\pi} : S \rightarrow \mathbb{R}$ that estimates the future rewards that are to be expected when starting in a specific state s and taking actions determined by policy π from then on. In this work, our goal is to learn a value function for the hassling problem that we shall represent using multilayer neural networks. If we assume to be in possession of an optimal state value function $V^{\star}$, it is easy to infer the corresponding optimal behavior policy by exploiting that value function greedily according to $\pi^{\star}(s) := \arg\max_{a \in A}\{r(s, a) + \sum_{s \in S} p(s, a, s') \cdot V^{\star}(s')\}$.

The Soccer Server [12] is a software that allows agents in the soccer simulation 2D League to play soccer in a client/server-based style: It simulates the playing field, communication, the environment and its dynamics, while the clients – eleven agents per team – are permitted to send their intended actions (e.g. a parameterized kick or dash command) once per simulation cycle to the server. Then, the server takes all agents' actions into account, computes the subsequent world state and provides all agents with information about their environment. Therefore, while the reward function is unknown to the agent, in soccer simulation the transition function p (model of the environment) is given since the way the Soccer Server simulates a soccer match is known. In the hassling task, however, the situation is aggravated: The presence of an adversary whose next actions cannot be controlled and hardly be predicted makes it impossible to accurately anticipate the successor state. Hence, only a rough approximation of p, that merely takes into account that part of the state that can be directly be influenced by the learning agent, is available.

3 Learning a Duel Behavior: The NeuroHassle Approach

"Aggressive playing" is a collocation frequently used in today's press 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, which is often considered to be crucial for a team's success.

3.1 Outline of the Learning Approach

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 players failed in conquering the ball in almost two thirds of all attempts – 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 adversary,
- 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. The state space for the problem at hand is continuous and high-dimensional; we restricted ourselves to 9 state dimensions:

- distance d between our player and the opponent ball leading player
- velocity (v_x and v_y component) of our player
- absolute value v_{opp} of the opponent's velocity
- position (b_x and b_y component) of the ball
- our player's body angle α relative to the opponent's position
- opponent player's body angle β relative to his direction towards our goal[1]
- 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

There are three important remarks to be made concerning that state representation. First, to simplify the state representation the center of the coordinate system is placed in the center of our player and the abscissa cuts through our and the opponent player. Yet, by the two features listed last, also the position of the hassling situation on the playing field is taken into account. Second, these nine state features do not fully represent the actual state (e.g. the adversary's

[1] Most dribbling players are heading towards the opponent team's goal when being allowed to dribble freely without interference.

exact velocity vector is unknown and cannot be inferred), yet the degree of partial observability is kept low. And third, the high dimensionality of the problem space requires the use of value function approximation mechanisms if we aim at applying value function-based reinforcement learning. To this end, we rely on multilayer perceptron neural networks.

The learning agent is allowed to use $dash(x)$ and $turn(y)$ commands where the domains of bots commands' parameters ($x \in [-100, 100]$, $y \in [-180°, 180°]$) are discretized such that in total 76 actions are available to the agent at each time step.

3.2 Training Situations and Training Regions

We designed a specialized set of *training situations* S for the learning agent ($|S| = 5000$) which is visualized in Figure 1. It basically consists of two semicircles across which the opponent ball leading player is placed randomly, whereas our learning player resides in the center. While the semicircle that lies in the direction towards our goal (defensive direction) has a radius of $3.0m$, the one in the opposite (offensive) direction is larger ($5.0m$). The intention behind that design of starting situations is that, on the one hand, the ball leading opponent typically starts immediately to dribble towards our goal, whereas the learning agent must interfere and try to hinder him from making progress. On the other hand, 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. Regarding the remaining state space dimensions, the ball is always randomly placed in the ball leading player's kickable area with zero velocity, and the velocities of both players as well as their body angles are chosen randomly as well.

Moreover, we defined four *training regions* on the playing field as sketched in Figure 2. 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

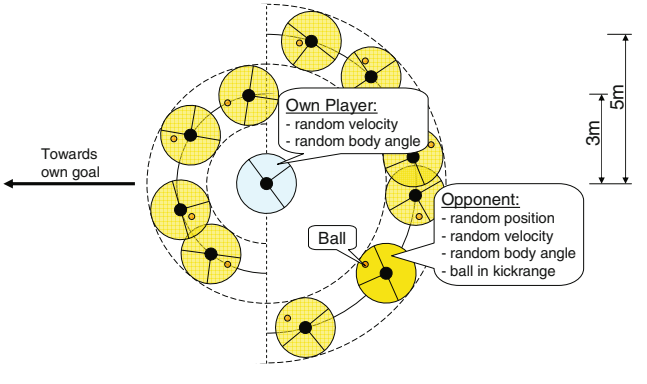

Fig. 1. General Training Scenario for Learning to Hassle Opponent Ball Leading Players

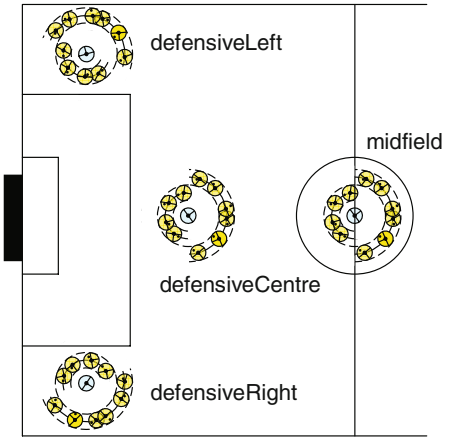

Fig. 2. Considered Sets of Training and Test Situations on the Playing Field

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. It is worth noting that the sets of situations used for the defensive left and right scenario are turned by approximately $\pm 70^\circ$ so that the smaller semicircle of starting positions for the ball leading opponent is oriented towards our goal.

3.3 A Note on Training Opponents

For training, of course, an adversary team is required. In the learning experiments we conducted, however, we did not employ an entire team, but just started a single player from each team involved[2]. We found that among the 16 published team binaries of RoboCup 2006 in Bremen, there were 2 binaries that were not functioning at all, 3 had severe problems (players crashing) with the player repositioning that necessarily must be made by the coach program during training, 6 were not usable for training (they feature dribbling behaviors that are seemingly inappropriate for the use during training, e.g. most of them kicked the ball just straight away), and 5 binaries seemed to be usable for our purposes. Since we preferred strong opponent teams with well developed dribbling capabilities, we additionally made use of two older champion binaries (teams STEP and UvA Trilearn in their version from 2005).

[2] Additionally, we started a goalkeeper as "support" for the learning player, since otherwise the ball leading player is sometimes mislead to shoot on the empty goal from a rather long distance.

3.4 Successful Episodes and Failures

A highly important issue concerns the question whether a single hassling episode, i.e. a single duel for the ball, was successful or not. Here, simply distinguishing between episodes during which the ball could be brought into the hassling player's kickable area or not, is not adequate – a more sophisticated differentiation is crucial for properly assessing the hassling capabilities acquired and is also essential for the use of a reinforcement learning approach.

After a careful analysis of the learning setting and of the way opponent ball leading players may behave during training, we found that the following five main outcomes of a training episode must be distinguished.

a) Erroneous Episode. A training episode can fail for many reasons. The dribbling player may lose the ball, the ball may go out of the field, a player may have not localized himself correctly and thus behave suboptimally, to name just a few. If any of these indications is perceived, then the corresponding episode is ended and abolished.

b) Success. A hassling episode can be considered successful, if the ball has been brought into the learning player's kickable area or if it has managed to position in such a manner that issuing a tackle command yields a successful tackle for the ball with very high probability.

c) Opponent Panic. It may also happen that the ball leading opponent player simply kicks the ball away (usually forwards), as soon as the learning agent has approached or hassled him too much, or if it simply considers his situation to be too hopeless to continue dribbling. Consequently, if an opponent issues such a kind of a panic kick, the episode under consideration may be regarded as a draw.

d) Failure. The hassling player fails, if none of the other cases has occurred. In particular, this means that the ball leading player has kept the ball in its kick range, or has even overrun the learning agent and escaped more than $7m$ from him, or has approached the goal such that a goal shot might become promising.

e) Time Out. We allocate a maximal episode duration of 35 time steps. If none of the cases mentioned before has occurred until that time, then the hassling episode is ended without a clear winner.

3.5 The Learning Algorithm

Temporal difference (TD) methods comprise a set of RL algorithms that incrementally update state value functions $V(s)$ after each transition (from state s to s') the agent has gone through. This is particularly useful when learning along trajectories $(s_0, s_1, \ldots, s_N)$ – as we do here – starting in some starting state s_0 and ending up in some terminal state $s_N \in G$. So, learning can be performed online, i.e. the processes of collecting (simulated) experience, and learning the value function run in parallel. Learning to hassle, we update the value function's estimates according to the $TD(1)$ update rule [13], 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.

We did also a number of tests concerning the usage of episodes that ended in some kind of a draw, i.e. in time-out episodes and in episodes where the adversary got into panic and kicked the ball straight away. Since an episode with a time-out is rather difficult to assess, we decided to not use those episodes during training. The probability of an opponent kicking the ball away when under high pressure, strongly depends on the respective opponent's dribbling behavior. For example, players from team TokyoTech frequently tend to perform panic kicks, even if the dribbling player is not yet under real pressure. Therefore, when training against such opponents it may be advisable to interpret panic kicks as failures, since, from the hassling player's perspective, the goal of bringing the ball into his control could not be achieved. As far as the learning results reported subsequently are concerned, we restricted ourselves to opponents with normal dribble behavior that do not panically and prematurely kick the ball away.

Due to the continuous, 9-dimensional state space to be covered a tabular representation of the state value function is impossible and instead the employment of a function approximator is necessary. For this task, we employ multi-layer perceptron neural networks with one hidden layer containing 18 neurons with sigmoidal activation function (hence, a 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. We do not discard old training instances and invoke a retraining of the neural network each time $|\tilde{S}|$ has been incremented by 250 instances. 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 efficient back-propagation variant *RPROP* [14].

4 Empirical Evaluation

To simplify training, the view restrictions imposed by the simulation environment (normally the Soccer Server provides agents only with information about objects in a restricted view cone) are turned off during learning, so that the learning agent obtains full state information.

4.1 Training to Hassle

Since the learning task is episodic, we do not use discounting. Further, we use a learning rate α of 1.0 and employ a purely greedy policy during training, since the neural network-based approximation of V in combination with the large sets

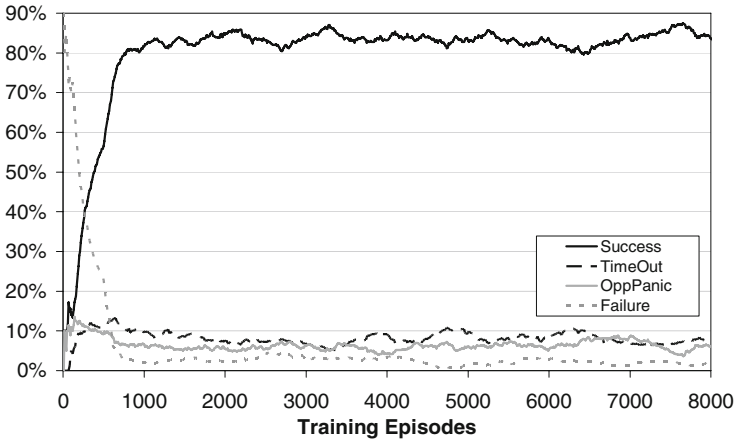

Fig. 3. Exemplary Learning Curve for Learning to Hassle (opponent during training: Wright Eagle, binary from RoboCup 2006)

of random episode start situations brings about a sufficient level of state space exploration. Figure 3 shows the learning curves for an example of a hassle learning experiment. Here, the neural network representing the value function from which a greedy policy is induced has been trained against the champion 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.

4.2 Testing the NeuroHassle Performance

Being trained against a single opponent team, the danger arises that the resulting behavior is over-specialized, does not generalize very well, and performs poorly against other teams. For this reason, we trained our hassling policy against a selection of teams and for different sets of training situations (on different places on the field). Having obtained a larger number candidate hassle behaviors in this way, it is in principle possible to utilize the corresponding highly specialized hassling policy depending on the current situation. E.g., when playing against UvA Trilearn and given that the ball leading opponent is positioned at $(-36, 24)$, it is most likely that the hassling policy performing best is the one that has been acquired when training against UvA for the *defensiveLeft* training set.

Apart from that, we are aware that the effectiveness of the policy obtained will be influenced by the presence of further opponent teammates. Then, of course, the ball leading opponent may behave differently, for example, it may play a pass quite a time before it is seriously hassled. However, if the defensive strategy is customized to support the intended aggressive hassling of the ball leader, and

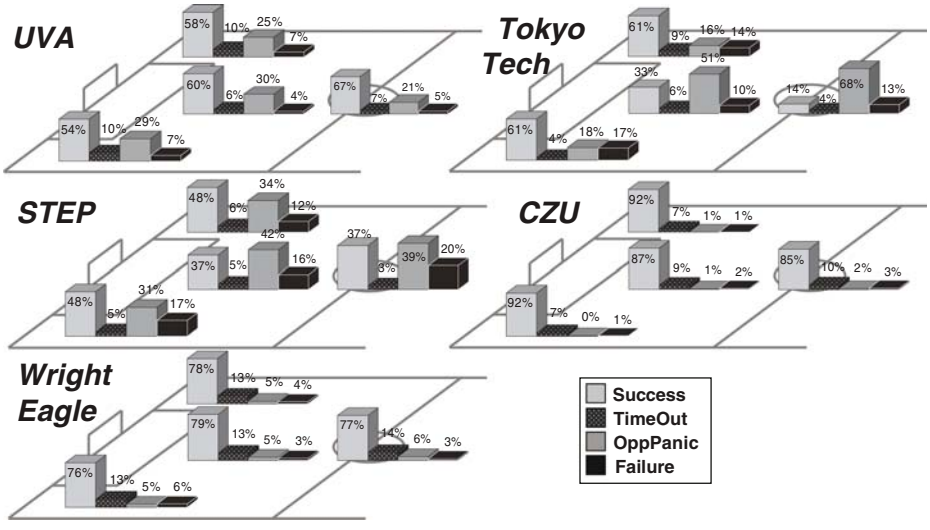

Fig. 4. Generalization Capabilities of the Acquired NeuroHassle Behavior against Various Opponent Teams and for Different Test Situation Sets

if it can be achieved that other adversary players are covered appropriately so that no passing opportunity arises (e.g. [8] focuses on that problem), then it is very likely that, during a real match, the ball leading player behaves similarly as during a training session.

In practice, unfortunately, the above-mentioned approach of employing a large ensemble of highly specialized policies (trained against various opponents and for varying position sets on the field, becomes quickly intractable: First, as discussed before (Section 3.3) it is impossible to acquire a specialized hassling behavior for every potential opponent since the binaries of most teams are not useful for training to hassle against them. And, of course, at times new teams appear for which no binary to be used as training partner is available at all. Second, each team changes from year to year. Hence, the risk of overfitting the hassling policy subject to a specific version of a certain team should not be underestimated.

Consequently, it has been our overall goal to create a hassling policy that generalizes very well over a number of teams (at least over those against which we could test and, in particular, against those teams that are still active). Figure 4 visualizes the performance of our NeuroHassle behavior when dueling against a selection of adversary teams and subject to test situations on different places on the playing field. Since a minimization of the share of failures is our main goal, one can conclude that the behavior acquired is quite effective. In any scenario considered the failure rates are not higher than 20%. Employing the learned NeuroHassle policy in our competition team *Brainstormers* was one of the crucial moves for winning the World Championships tournament RoboCup 2007 in Atlanta.

4.3 Integration into the Competition Team

Designing a strong team defense strategy represents an extremely challenging (and fragile) task when developing a soccer simulation team. It is well-known that in a defensive situation, i.e. when the ball is under control of the opponent team, the players' mission is to either (a) cover opponent players and block potential pass ways, or (b) to try to conquer the ball from the ball leader while simultaneously hindering him from dribbling ahead. Moreover, both tasks have to be assigned to the defending team's players in such a manner that no conflicts arise (e.g. no two players decide for hassling the ball leader in parallel while another opponent remains uncovered) and such that collaborative defense utility is maximized. Generally, it is advisable to select the player closest to the ball leading opponent for executing task (b), whereas the remaining players perform task (a), although a number of exceptional situations exist. A thorough discussion of this issue as well as of how to best solve task (a) is beyond the scope of this paper (see, for example, [8] for more details on these points).

We finish this section with providing some numbers on the utilization of the NeuroHassle policy during actual games against strong opponent teams. During the run of a standard game (6000 time steps), the players of our team start on average 66 hassling episodes. Therefore, even under the very conservative assumption that only about half of all hassle attempts are successful, we can draw the conclusion that the NeuroHassle behavior allows for conquering the ball at least 30 times per game. Furthermore, we determined that on average an opponent ball leading player is being hassled by one of our players during approximately 41.2% of the time he is in ball possession (ball within kick range). This corresponds to a NeuroHassle usage share of circa 14.8% in defensive situations during which the opponent team is building up its attack.

5 Summary

In this paper, we have reported on a comprehensive case study in RoboCup simulated soccer that aims at acquiring a defensive behavior which is meant to disturb an opponent ball leader and to reconquer the ball from him – a capability which is of substantial importance for team success. We developed a comprehensive set of training scenarios and pursued a reinforcement learning method based on neural value function approximation to obtain a good hassle policy for our *Brainstormers* soccer simulation 2D team. The empirical results of applying the acquired policy against various opponent teams were highly encouraging. Consequently, we also successfully applied the described NeuroHassle behavior during 2007's RoboCup World Championships tournament.

Generally, prior to new tournaments a retraining of the NeuroHassle policy must be performed so that it is up-to-date with recent changes introduced in opponent teams' strategies. Apart from that, in future work we also plan to let the agent train against different training opponents that are exchanged from time to time (during a single training session) so that the generalization capabilities can be further increased.

References

1. Kitano, H., Asada, M., Kuniyoshi, Y., Noda, I., Abd, H., Matsubara, E.O.: RoboCup: A Challenge Problem for AI. AI Magazine 18, 73–85 (1997)
2. Kyrylov, V., Greber, M., Bergman, D.: Multi-Criteria Optimization of Ball Passing in Simulated Soccer. Journal of Multi-Criteria Decision Analysis 13, 103–113 (2005)
3. Stone, P., Sutton, R., Kuhlmann, G.: Reinforcement Learning for RoboCup-Soccer Keepaway. Adaptive Bahvior 13, 165–188 (2005)
4. Dashti, H., Aghaeepour, N., Asadi, S., Bastani, M., Delafkar, Z., Disfani, F., Ghaderi, S., Kamali, S.: Dynamic Positioning Based on Voronoi Cells (DPVC). In: Bredenfeld, A., Jacoff, A., Noda, I., Takahashi, Y. (eds.) RoboCup 2005. LNCS, vol. 4020, pp. 219–229. Springer, Heidelberg (2006)
5. Reis, L., Lau, N., Oliveira, E.: Situation Based Strategic Positioning for Coordinating a Team of Homogeneous Agents. In: Hannebauer, M., Wendler, J., Pagello, E. (eds.) ECAI-WS 2000. LNCS, vol. 2103, pp. 175–197. Springer, Heidelberg (2001)
6. 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, pp. 68–75. IEEE Press, Los Alamitos (2007)
7. Kalyanakrishnan, S., Liu, Y., Stone, P.: Half Field Offense in RoboCup Soccer: A Multiagent Reinforcement Learning Case Study. In: Lakemeyer, G., Sklar, E., Sorrenti, D.G., Takahashi, T. (eds.) RoboCup 2006: Robot Soccer World Cup X. LNCS, vol. 4434, pp. 72–85. Springer, Heidelberg (2007)
8. Kyrylov, V., Hou, E.: While the Ball in the Digital Soccer Is Rolling, Where the Non-Player Characters Should Go in a Defensive Situation?. In: Proceedings of Future Play, Toronto, Canada, pp. 13–17 (2007)
9. Sutton, R.S., Barto, A.G.: Reinforcement Learning. An Introduction. MIT Press/A Bradford Book, Cambridge (1998)
10. Gabel, T., Riedmiller, M.: Learning a Partial Behavior for a Competitive Robotic Soccer Agent. KI Zeitschrift 20, 18–23 (2006)
11. Bertsekas, D.P., Tsitsiklis, J.N.: Neuro Dynamic Programming. Athena Scientific, Belmont, USA (1996)
12. Noda, I., Matsubara, H., Hiraki, K., Frank, I.: Soccer Server: A Tool for Research on Multi-Agent Systems. Applied Artificial Intelligence 12, 233–250 (1998)
13. Sutton, R.S.: Learning to Predict by the Methods of Temporal Differences. Machine Learning 3, 9–44 (1988)
14. Riedmiller, M., Braun, H.: A Direct Adaptive Method for Faster Backpropagation Learning: The RPROP Algorithm. In: Ruspini, H. (ed.) Proceedings of the International Conference on Neural Networks (ICNN), San Francisco, pp. 586–591 (1993)