Progress in Learning 3 vs. 2 Keepaway

Gregory Kuhlmann and Peter Stone

Department of Computer Sciences
The University of Texas at Austin
Austin, Texas 78712-1188
{kuhlmann,pstone}@cs.utexas.edu
http://www.cs.utexas.edu/~{kuhlmann,pstone}

Abstract. Reinforcement learning has been successfully applied to several subtasks in the RoboCup simulated soccer domain. *Keepaway* is one such task. One notable success in the keepaway domain has been the application of SMDP Sarsa(λ) with tile-coding function approximation [9]. However, this success was achieved with the help of some significant task simplifications, including the delivery of complete, noise-free world-state information to the agents. Here we demonstrate that this task simplification was unnecessary and further extend the previous empirical results on this task.

1 Introduction

RoboCup simulated soccer has been a popular test-bed for studying reinforcement learning algorithms over the years. In principle, modern reinforcement learning methods are reasonably well suited to meeting the challenges of RoboCup simulated soccer; and RoboCup soccer is a large and difficult instance of many of the issues which have been addressed in small, isolated cases in previous reinforcement learning research. Despite substantial previous work (e.g., [10, 7]), the extent to which modern reinforcement learning methods can meet these challenges remains an open question.

This article builds upon the work of Stone & Sutton [9] who began scaling reinforcement learning up to RoboCup simulated soccer by considering a subtask of soccer involving fewer than the full 22 players. In particular, they consider the task of *keepaway*, a subproblem of RoboCup soccer in which one team, the *keepers*, tries to maintain possession of the ball within a limited region, while the opposing team, the *takers*, attempts to gain possession. Parameters of the task include the size of the region, the number of keepers, and the number of takers. We have recently incorporated the framework for this domain into the standard, open-source RoboCup soccer simulation software [4].

In their previous work, Stone & Sutton [9] apply episodic SMDP Sarsa(λ) with linear tile-coding function approximation (CMACs [1]) to the keepaway task. The learners choose not from the simulator's primitive actions (e.g. kick, dash, and turn) but from higher level actions constructed from a set of basic skills (implemented by the CMUnited-99 team [8]).

Keepers have the freedom to decide which action to take only when in possession of the ball. A keeper in possession may either hold the ball or pass to one of its teammates. Keepers not in possession of the ball are required to select the **Receive** option in which the fastest player to the ball goes to the ball and the remaining players try to get open for a pass.

The keepers' set of state variables are computed based on the positions of: the keepers K_1–K_n and takers T_1–T_m, ordered by increasing distance from K_1; and C, the center of the playing region. Let $dist(a, b)$ be the distance between a and b and $ang(a, b, c)$ be the angle between a and c with vertex at b. For 3 keepers and 2 takers, we used the following 13 state variables: $dist(K_1, C)$; $dist(K_2, C)$; $dist(K_3, C)$; $dist(T_1, C)$; $dist(T_2, C)$; $dist(K_1, K_2)$; $dist(K_1, K_3)$; $dist(K_1, T_1)$; $dist(K_1, T_2)$; $\mathrm{Min}(dist(K_2, T_1), dist(K_2, T_2))$; $\mathrm{Min}(dist(K_3, T_1), dist(K_3, T_2))$; $\mathrm{Min}(ang(K_2, K_1, T_1), ang(K_2, K_1, T_2))$; $\mathrm{Min}(ang(K_3, K_1, T_1), ang(K_3, K_1, T_2))$.

The behavior of the takers is relatively simple. The two fastest takers to the ball go to the ball while the remaining takers try to block open passing lanes.

Using this setup, Stone & Sutton [9] were able to show an increase in average episode duration over time when keepers learned against hand-coded takers. They compared their results with a **Random** policy that chooses among its options with uniform probability, an **Always Hold** policy, and a hand-coded policy that uses a decision tree for pass evaluation. Experiments were conducted on several different field sizes. In each case, the keepers were able to learn policies that outperformed all of the benchmarks. Most of their experiments matched 3 keepers against 2 takers. However, they also showed that their results extend to the 4 vs. 3 scenario.

In the RoboCup soccer simulator, agents typically have limited and noisy sensors: each player can see objects within a 90^o view cone, and the precision of an object's sensed location degrades with distance. However, to simplify the task, Stone & Sutton [9] removed these restrictions. The learners were given 360^o of noiseless vision. Here, we demonstrate that these simplifications are unnecessary: the agents are able to learn successful policies despite having sensor noise and limited vision. We also extend the results to larger teams and provide further insights into the previous results based on additional controlled experiments. One of our key observations is that a large source of the problem difficulty is the fact that multiple agents learn simultaneously: when a single agent learns in the presence of pre-trained teammates, it is able to do so significantly more quickly.

2 Experimental Setup and Results

This section addresses each of the following questions with focused experiments in the keepaway domain:

1. Does the learning approach described above continue to work if the agents are limited to noisy, narrowed vision?
2. How does a learned policy perform in comparison to a hand-coded policy that has been manually tuned?
3. How robust are these policies to differing field sizes?
4. How dependent are the results on the state representation?
5. How well do the results scale to larger problems?
6. Is the source of the difficulty the learning task itself, or the fact that multiple agents are learning simultaneously?

2.1 Limited Vision

Limited vision introduces two challenges with respect to the complete vision setup of Stone & Sutton. First, the agents must model their uncertainty in the world state and make the appropriate decisions based on those uncertainties. Second, the agents must occasionally take explicit information-gathering actions to increase their confidence in the world state.

To model state uncertainty, a player stores a certainty factor along with each state variable that decays over time. When the player receives sensory information, it updates its world state and resets the certainty factor. If the keeper with the ball does not have reliable information about the position of its teammates, then we force the player to hold the ball and turn its neck until it has enough information to make a pass.

Using this method, we attempted to reproduce the results of Stone & Sutton for 3 vs. 2 keepaway on a $20m \times 20m$ field but without the simplification of unrestricted vision. In their work, keepers were able to learn policies with average episode durations of around 15 seconds. However, learning with noisy, narrowed vision is a more difficult problem than learning with complete knowledge of the state. For this reason, we expected our learners to hold the ball for less time. However, since these same difficulties impact the benchmark policies, the salient question is whether or not learning is still able to outperform the benchmarks.

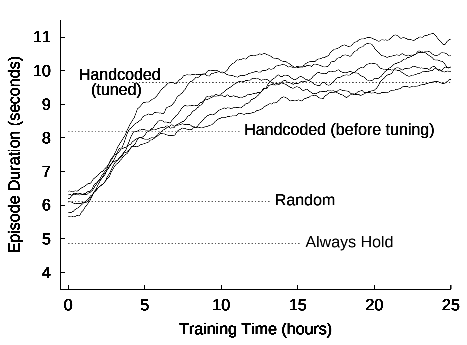

Fig. 1. Learning curves and benchmarks for limited vision: 3 keepers vs. 2 takers.

We ran a series of 6 independent learning trials in which the keepers learned while playing against the hand-coded takers. In each run, the keepers gradually improved their performance before leveling off after about 25 hours of simulator time. The learning curves are shown in Figure 1. We plotted all 6 trials to give a sense of the variance.

All of the learning runs were able to outperform the **Always Hold** and **Random** benchmark policies. The learned policies also outperformed our **Hand-coded** policy which we describe in the next section.

2.2 Comparison to Hand-coded

In addition to the **Always Hold** and **Random** benchmark policies described previously, we compared our learners to a new **Hand-coded** policy. In this policy, the keeper in possession, K_1 assigns a score to each of its teammates based on how "open" they are. The degree to which the player is open is calculated as a linear combination of the teammate's distance to its nearest opponent, and the angle between the teammate, K_1, and the opponent closest to the passing lane. The relative importance of these two features are weighted by the coefficient α.

If the most open teammate has a score above the threshold, β, then K_1 will pass to this player. Otherwise, K_1 will hold the ball for one cycle.

This **Hand-coded** policy was designed to use only state variables and calculations that are available to the learner. We chose initial values for α and β based on educated guesses. We tuned these values by experimenting with values near our initial guesses. Altogether, we tried about 30 combinations, before settling on our final tuned values.

We ran a few thousand episodes of our tuned **Hand-coded** policy and found that it was able to keep the ball for an average of 9.6 seconds per episode. Also, for comparison, we tested our **Hand-coded** policy before manual tuning. This policy was able to hold the ball for an average of 8.2 seconds. From Figure 1 we can see that the keepers are able to learn policies that outperform our initial **Hand-coded** policy and exhibit performance roughly as good as (perhaps slightly better than) the tuned version. We examined the **Hand-coded** policy further to find out to what degree its performance is dependent on tuning.

2.3 Robustness to Differing Field Sizes

Stone & Sutton already demonstrated that learning is robust to changes in field sizes [9]. Here we verify that learning is still robust to such changes even with the addition of significant state uncertainty. We also benchmark these results against the robustness of the **Hand-coded** policy to the same changes. Overall, we expect that as the size of the play region gets smaller, the keepers will have a harder time maintaining possession of the ball regardless of policy. Here we compare the **Hand-coded** policy to learned policies on five different field sizes. The average episode durations for both solutions are shown in Figure 2. Each value for the learned runs was calculated as an average of six separately learned policies.

	Keeper Policy	
Field Size	Hand-coded	Learned ($\pm 1\sigma$)
30×30	**19.8**	18.2 ± 1.1
25×25	**15.4**	14.8 ± 0.3
20×20	9.6	$\mathbf{10.4 \pm 0.4}$
15×15	6.1	$\mathbf{7.4 \pm 0.9}$
10×10	2.7	$\mathbf{3.7 \pm 0.4}$

Fig. 2. Average possession times (in simulator seconds) for hand-coded and learned policies on various field sizes.

As can be seen from the table, the hand-coded policy does better on the easier problems ($30m \times 30m$ and $25m \times 25m$), but the learned policies do better on the more difficult problems.

A possible explanation for this result is that the easier cases of keepaway have more intuitive solutions. Hence, these problems lend themselves to a hand-coded approach. However, without any impetus to choose a simpler approach, learned policies tend to be more asymmetric and irregular. This lack of rhythm seems to lead to suboptimal performance on easier tasks.

In contrast, when the keepers are forced to play in a smaller area, the "intuitive" solution breaks down. The hand-coded keepers tend to pass too frequently, leading to missed passes. In these more difficult tasks, the trained keepers appear to find "safer" solutions in which the ball is held for longer periods of time. This approach leads to fewer missed passes and better overall performance than the hand-coded solution.

2.4 Changing the State Representation

A frequent challenge in machine learning is finding the correct state representation. In all of the experiments reported so far, we have used the same state variables as in Stone & Sutton's work, which were chosen without any detailed exploration [9]. Here we explore how sensitive the learning is to the set of state variables used.

As a starting point, notice that our **Hand-coded** policy uses only a small subset of the 13 state variables mentioned previously. Because the **Hand-coded** policy did quite well without using the remaining variables, we wondered if perhaps the unused state variables were not essential for the keepaway task.

To test this theory, we performed a series of learning runs in which the keepers used only the five variables from the hand-coded policy. Figure 3 shows the learning curves for six runs. As is apparent from the graph, the results are very similar to those in Figure 1. Although we found that the keepers were able to achieve better than random performance with as little as one state variable, the five variables used in the hand-coded policy seem to be minimal for peak performance. Notice by comparing Figures 1 and 3 that the keepers are able to learn at approximately the same rate whether the nonessential state variables are present or not.

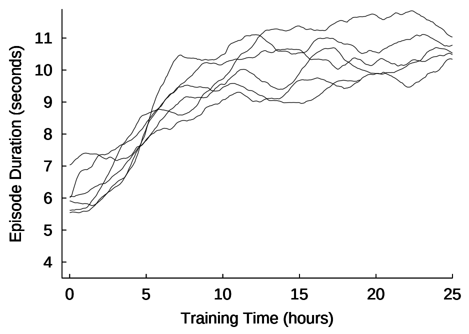

Fig. 3. Learning with only the 5 state variables from the **Hand-coded** policy.

To explore this notion further, we tried adding additional state variables to the original 13. We ran two separate experiments. In the first experiment, we added 2 new angles that appeared relevant but perhaps redundant. $ang(K_1, C, K_2)$; $ang(K_1, C, K_3)$. In the second experiment, we added 2 completely irrelevant variables: each time step, new values were randomly chosen from $[-90,90]$ with uniform probability.

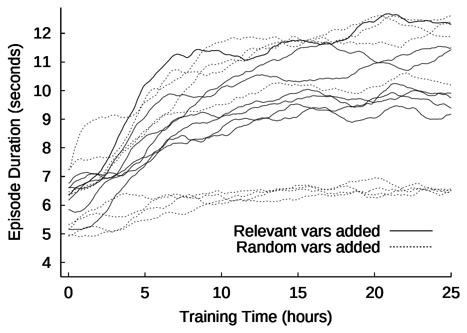

Fig. 4. Learning with the original 13 state variables plus an additional two.

From Figure 4, we can see that the learners are not greatly affected by the addition of relevant variables. The learning curves look roughly the same as the ones that used the original 13 state variables (Figure 1). However, the curves corresponding to the additional random variables look somewhat different. The curves can clearly be divided into two groups. In the first group, teams are able to perform about as well as the ones that used the original 13 variables. In the second group, the agents perform very poorly. It appears that agents in the second group are confused by the irrelevant variables while the agents in the first group are not. This distinction seems to be made in the early stages of learning (before the 1000th episode corresponding to the first data point on the graph). The learning curves that start off low stay low. The ones that start off high continue to ascend.

From these results, we conclude that it is important to choose relevant variables for the state representation. However, it is unnecessary to carefully choose the minimum set of these variables.

2.5 Scaling to Larger Problems

In addition to our experiments with 3 vs. 2 keepaway, we ran a series of trials with larger team sizes to determine how well our techniques scale. First we performed several learning runs with 4 keepers playing against 3 hand-coded takers. We compared these to our three benchmark policies. The results are shown in Figure 5. As in the 3 vs. 2 case, the players are able to learn policies that outperform all of the benchmarks.

We also ran a series of experiments with 5 vs. 4 keepaway. The learning curves for these runs along with our three benchmarks are shown in Figure 6. Again, the learned policies outperform all benchmarks. As far as the authors are aware, these experiments represent the largest scale keepaway problems that have been successfully learned to date.

From these graphs, we see that the learning time approximately doubles every time we move up in size. In 3 vs. 2, the performance plateaus after roughly (by eyeballing the graphs) 15 hours of training. In 4 vs. 3, it takes about 30 hours to learn. In 5 vs. 4, it takes about 70 hours.

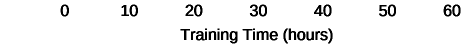

Fig. 5. Training 4 Keepers against 3 takers with benchmarks.

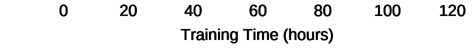

Fig. 6. Training 5 Keepers against 4 takers with benchmarks.

2.6 Difficulty of Multiagent Learning

A key outstanding question about keepaway is whether it is difficult as an individual learning task, or if the multiagent component of the problem is the largest source of difficulty. To see how the number of agents learning simultaneously affects the overall training time, we ran a series of experiments in which a subset of the keepers learned while the remaining teammates followed a fixed policy learned previously. We ran each experiment three times. The learning curves for all nine runs are shown in Figure 7.

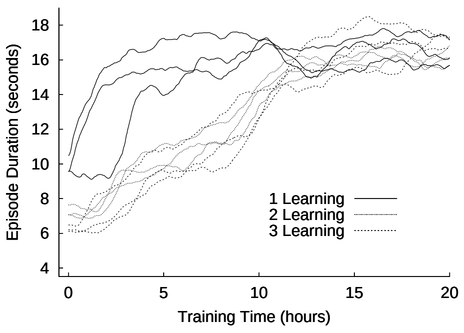

Fig. 7. Learning curves for varying number of keepers learning simultaneously.

From the graph we can see that the learning curves for 2 learning agents and 3 learning agents look roughly the same. However, the runs with only 1 player learning peak much sooner. Apparently, having pre-trained teammates allows an agent to learn much faster. However, if more than one keeper is learning, the presence of a pre-trained teammate is not helpful. This result suggests that multiagent learning is an inherently more difficult problem than single agent learning, at least for this task. In the long run, all three configurations' learned policies are roughly equivalent. The number of learning agents does not seem to affect the quality of the policy, only the rate at which the policy is learned.

3 Related Work

Several previous studies have used keepaway soccer as a machine learning testbed. Whiteson & Stone [11] used neuroevolution to train keepers in the SoccerBots domain [3]. The players were able to learn several conceptually different tasks from basic skills to higher-level reasoning using a hierarchical approach they call "concurrent layered learning." The keepers were evaluated based on the number of completed passes. Hsu & Gustafson [5] evolved keepers for 3 vs. 1 keepaway in the much simpler and more abstract TeamBots simulator [2]. Keepers were trained to minimize the number of turnovers in fixed duration games. It is difficult to compare these approaches to ours because they use different fitness functions and different game dynamics.

More comparable work to ours applied evolutionary algorithms to train 3 keepers against 2 takers in the RoboCup soccer simulator [6]. Similar to our work, they focused on learning keepers in possession of the ball. The keepers chose from the same high-level behaviors as ours. Also, they used average episode duration to evaluate keeper performance. However, because their high-level behaviors and basic skills were implemented independently from ours, it is difficult to compare the two learning approaches empirically. Additional related work is discussed in [9].

4 Conclusion and Future Work

Taken together, the results reported in this paper show that SMDP Sarsa(λ) with tile-coding scales further and is more robust than has been previously shown. Even in the face of significant sensor noise and hidden state, it achieves results at least as good as those of a tuned hand-coded policy.

The main contribution of this paper is a deeper understanding of the difficulties of scaling up reinforcement learning to RoboCup soccer. We focused on the keepaway task and demonstrated that players are able to improve their performance despite having noisy, narrowed vision. We also introduced a new hand-coded policy and compared it for robustness to our learned policies. We demonstrated the difficulty of scaling up current methods and provided evidence that this difficulty arises mainly out of the fact that several agents are learning simultaneously.

References

1. J. S. Albus. *Brains, Behavior, and Robotics.* Byte Books, Peterborough, NH, 1981.
2. Tucker Balch. Teambots, 2000. `http://www.teambots.org`.
3. Tucker Balch. Teambots domain: Soccerbots, 2000. `http://www-2.cs.cmu.edu/~trb/TeamBots/Domains/SoccerBots`.
4. RoboCup Soccer Simulator Maintenance Group. The robocup soccer simulator, 2003. `http://sserver.sourceforge.net`.
5. W. H. Hsu and S. M. Gustafson. Genetic programming and multi-agent layered learning by reinforcements. In *Genetic and Evolutionary Computation Conference*, New York,NY, July 2002.
6. Anthony Di Pietro, Lyndon While, and Luigi Barone. Learning in RoboCup keepaway using evolutionary algorithms. In W. B. Langdon, E. Cantú-Paz, K. Mathias, R. Roy, D. Davis, R. Poli, K. Balakrishnan, V. Honavar, G. Rudolph, J. Wegener, L. Bull, M. A. Potter, A. C. Schultz, J. F. Miller, E. Burke, and N. Jonoska, editors, *GECCO 2002: Proceedings of the Genetic and Evolutionary Computation Conference*, pages 1065–1072, New York, 9-13 July 2002. Morgan Kaufmann Publishers.
7. M. Riedmiller, A. Merke, D. Meier, A. Hoffman, A. Sinner, O. Thate, and R. Ehrmann. Karlsruhe brainstormers—a reinforcement learning approach to robotic soccer. In Peter Stone, Tucker Balch, and Gerhard Kraetszchmar, editors, *RoboCup-2000: Robot Soccer World Cup IV.* Springer Verlag, Berlin, 2001.
8. Peter Stone, Patrick Riley, and Manuela Veloso. The CMUnited-99 champion simulator team. In M. Veloso, E. Pagello, and H. Kitano, editors, *RoboCup-99: Robot Soccer World Cup III*, pages 35–48. Springer Verlag, Berlin, 2000.
9. Peter Stone and Richard S. Sutton. Scaling reinforcement learning toward RoboCup soccer. In *Proceedings of the Eighteenth International Conference on Machine Learning*, pages 537–544. Morgan Kaufmann, San Francisco, CA, 2001.
10. Peter Stone and Manuela Veloso. Team-partitioned, opaque-transition reinforcement learning. In Minoru Asada and Hiroaki Kitano, editors, *RoboCup-98: Robot Soccer World Cup II.* Springer Verlag, Berlin, 1999. Also in *Proceedings of the Third International Conference on Autonomous Agents*, 1999.
11. Shimon Whiteson and Peter Stone. Concurrent layered learning. In *Second International Joint Conference on Autonomous Agents and Multiagent Systems*, July 2003.