Advances in Complex Systems

An Empirical Study of Potential-Based Reward Shaping and Advice in Complex, Multi-Agent Systems

Sam Devlin, Marek Grześ and Daniel Kudenko
University of York, UK
{*devlin, kudenko*}*@cs.york.ac.uk*
University of Waterloo, CA
mgrzes@cs.uwaterloo.ca

Received Sept 30th 2010
Revised Feb 21st 2011
Accepted Feb 21st 2011

This paper investigates the impact of reward shaping in multi-agent reinforcement learning as a way to incorporate domain knowledge about good strategies. In theory, potential-based reward shaping does not alter the Nash Equilibria of a stochastic game, only the exploration of the shaped agent. We demonstrate empirically the performance of reward shaping in two problem domains within the context of RoboCup KeepAway by designing three reward shaping schemes, encouraging specific behaviour such as keeping a minimum distance from other players on the same team and taking on specific roles. The results illustrate that reward shaping with multiple, simultaneous learning agents can reduce the time needed to learn a suitable policy and can alter the final group performance.

Keywords: Reinforcement Learning; Multi-Agent; Reward Shaping.

1. Introduction

Multi-Agent Systems (MAS) are becoming increasingly popular because in many practical applications it is natural to model the environment with multiple agents or decomposing domains which are inherently single-agent into multiple agents may allow for more efficient solutions [37]. One of the methods of designing intelligent agents is the use of machine learning to implement adaptive, autonomous, and self-improving behaviour. Reinforcement learning in particular represents a natural fit to learn adaptive behaviour in a multi-agent scenario.

Whilst reinforcement learning can deal with problems with combinatorial huge state spaces in a fully observable setting [22, 28], the multi-agent scenario is a bigger challenge [5]. The most significant problem is other agents, which execute their own actions and subsequently influence the state of the world. This makes the problem partially observable because of the uncertainty in the behaviour of other agents and also non-stationary because other agents may concurrently learn and improve their behaviour. Additionally, the state-action space of a MAS grows exponentially with the number of agents, which may considerably slow down convergence.

Most existing reinforcement learning algorithms were proposed under the assumption that there is no knowledge available about the problem or the Markov Decision Process (MDP) in particular. This is, however, often not the case in many practical applications. In many domains, heuristic knowledge can be easily identified by the designer of the system [24] or acquired using reasoning or learning [12]. In the area of single-agent reinforcement learning, potential-based reward shaping has been proven to be a principled and theoretically correct method of incorporating heuristic knowledge into an agent [21]. To date, only relatively simple multi-agent scenarios have been studied with regard to potential-based reward shaping [1, 17]. The contribution of our paper is an application specifically to complex MAS. In this work we focus on two distinct learning tasks within the RoboCup domain; a popular soccer/football simulator. The paper explains a method to incorporate prior domain knowledge into multiple agents learning in the same environment and demonstrates the implications of doing so.

Our empirical evaluation is based on the RoboCup domain for two reasons. Firstly, RoboCup is an international project (see Section 4 for details) which has been proven to provide an experimental framework in which various technologies can be integrated and evaluated. The second reason why we focus on the RoboCup domain is that our aim in this project is to investigate knowledge-based, multi-agent reinforcement learning approaches. This requires a well-defined and challenging domain, such as RoboCup, where domain specific knowledge can be identified.

In our experiments we investigate three types of multi-agent knowledge:

(1) How agents should maintain states relative to each other (e.g. keep a minimum distance).
(2) How role specialisation can improve overall performance by explicitly encouraging heterogeneous behaviours in a multi-agent team (e.g. specialising in tackling the ball-controlling opponent).
(3) The combination of (1) and (2).

These three types of knowledge were chosen as they cover examples of state-based and action-based knowledge. This is sufficient to allow us to experiment with both Ng's potential-based reward shaping [21] and Wiewiora et al.'s potential-based advice [9] both alone and in combination. We do not propose that this is an exhaustive list of knowledge types applicable to multi-agent reinforcement learning but do believe they are generally applicable to most MAS.

In this work, we show that reward shaping in multi-agent reinforcement learning can speed up learning, as it does in single agent, but, unlike single agent, can alter the policy converged to. These results demonstrate that reward shaping can be successfully used in multi-agent reinforcement learning to incorporate domain knowledge and encourage specific behaviours. Provided a good heuristic, these behaviours will direct the learning towards convergence on a better final performance. However, a poor heuristic can also detract from performance.

Since the full game of soccer is complex, researchers developed several simulated

environments which can be used to evaluate techniques for specific sub-problems. One of these sub-problems is the KeepAway task [27]. There are two teams of agents in this domain: a team of keepers which learn how to maintain possession of the ball and a team of takers which learn how to get the ball. In this paper, experiments on learning the behaviours of both RoboCup keepers and takers are presented separately.

The paper is organised as follows. Section 2 presents a more detailed introduction to reinforcement learning and Section 3 introduces reward shaping. The subsequent section introduces RoboCup Soccer, KeepAway and TakeAway; the chosen problem domains. Next, Sections 5 and 6 discuss our approach to learning keepers and takers respectively both with and without reward shaping. The final section concludes the paper, making comments about the general applicability of these methods and the benefits of their use.

2. Reinforcement Learning and Markov Decision Problems

Reinforcement learning is a paradigm which allows agents to learn by reward and punishment from interactions with the environment [30]. The numeric feedback received from the environment is used to improve the agent's actions. The majority of work in the area of reinforcement learning applies a Markov Decision Process (MDP) as a mathematical model [23].

An MDP is a tuple $\langle S, A, T, R \rangle$, where S is the state space, A is the action space, $T(s, a, s') = Pr(s'|s, a)$ is the probability that action a in state s will lead to state s', and $R(s, a, s')$ is the immediate reward r received when action a taken in state s results in a transition to state s'. The problem of solving an MDP is to find a policy (i.e., mapping from states to actions) which maximises the accumulated reward. When the environment dynamics (transition probabilities and a reward function) are available, this task can be solved using iterative approaches like policy and value iteration [3].

When the environment dynamics are not available (as with most true environments) value iteration cannot be used. However, the concept of an iterative approach remains the backbone of the majority of reinforcement learning algorithms. These algorithms apply so called temporal-difference updates to propagate information about values of states, $V(s)$, or state-action pairs, $Q(s, a)$ [29]. These updates are based on the difference of the two temporally different estimates of a particular state or state-action value. The SARSA algorithm is such a method [30]. After each real transition, $(s, a) \rightarrow (s', r)$, in the environment, it updates state-action values by the formula:

$$Q(s, a) \leftarrow Q(s, a) + \alpha[r + \gamma Q(s', a') - Q(s, a)] \quad (1)$$

where α is the rate of learning and γ is the discount factor. It modifies the value of taking action a in state s, when after executing this action the environment returned reward r, moved to a new state s', and action a' was chosen in state s'.

Problem domains modelled in an MDP only allow for actions of equal time to perform, they must all complete in one time step. This is often too simple a model as in many simulated environments some actions take longer than others. In such cases a more useful model is a Semi-Markov Decision Process (SMDP). An SMDP replaces the set of actions in the MDP tuple $< S, A, T, R >$ with a set of macro-actions. Macro-actions may take one or more time steps to complete. When chosen by an agent, the environment does not provide a reward or prompt for a new macro-action until the last has completed. The temporal difference in macro-actions must be considered when implementing the reward function because, dependant on the intended goal, shorter macro-actions may be preferred to longer macro-actions.

Another important implementation consideration is that of balancing exploration and exploitation. As agents learn in the environment it is important for them to exploit state-action pairs they have learnt are rewarded well, but to find any such pairs they must initially explore. Later on exploration allows the discovery of state-action pairs not previously considered but of higher reward. If exploration is stopped too early it can cause an agent to get stuck in a local optimum, not noticing the better policies because they are greedily sticking to what they have already learnt.

A common method of balancing exploration and exploitation is $\epsilon-$greedy. This method keeps a constant amount of exploration throughout learning. With probability ϵ the agent explores by choosing a random action and with probability $1 - \epsilon$ the agent exploits its current knowledge and chooses the highest value action for the current state. The one parameter, ϵ, is an important setting in agents using this method.

2.1. *Multi-Agent Reinforcement Learning*

Applications of reinforcement learning to MAS typically take one of two approaches; multiple individual learners or joint action learners [6]. The latter is a group of multi-agent specific algorithms designed to consider the existence of other agents. The former is the deployment of multiple agents each using a single-agent reinforcement learning algorithm.

Multiple individual learners assume any other agents to be a part of the environment and so, as the others simultaneously learn, the environment appears to be dynamic as the probability of transition when taking action a in state s changes over time. To overcome the appearance of a dynamic environment, joint action learners were developed that extend their value function to consider for each state the value of each possible combination of actions by all agents.

Learning by joint action, however, breaks a common fundamental concept of MAS in which each agent is self-motivated and so may not consent to the broadcasting of their action choices. Furthermore, the consideration of the joint action causes an exponential increase in the number of values that must be calculated with each additional agent added to the system. Typically, joint action learning al-

gorithms have only been demonstrated in trivial problem domains [34, 13, 6] whilst applications in MAS most often implement multiple individual learners [18, 31, 32]. For these reasons, this work will focus on multiple individual learners and not joint action learners. However, it is expected that the application of these approaches to joint action learners would have similar benefits.

Unlike single-agent reinforcement learning, where the goal is obviously to maximise the individual's reward, when multiple self-motivated agents are deployed not all agents can receive their maximum reward all of the time. Instead some compromise must be made and so the agents typically aim to converge to a Nash Equilibrium (N.E.)[20].

N.E. is a popular concept from the field of game theory. A game, in this context, is any interaction between two or more agents and is, therefore, a suitable model of many MAS. In a N.E., the action chosen by each agent is the best response to the actions of all other agents. All games have at least one N.E., provided there are only a finite number of players and actions. Some, however, will have more than one N.E. and they may not be valued the same. The optimal N.E. is an ambiguous term, but in team games typically refers to the N.E. of highest global utility (the sum value of all agents' rewards). [4, 10]

In multi-agent reinforcement learning, a N.E. is a joint policy where each agent's individual policy is the best response to the joint policy of all other agents. A N.E. is not the only possible goal of multi-agent reinforcement learning, but is the most common [25].

3. Reward Shaping

The immediate reward r, which is in the update rule given by Equation 1, represents the feedback from the environment. The idea of *reward shaping* is to provide an additional reward which will improve the convergence of the learning agent with regard to the learning speed [21, 24]. This concept can be represented by the following formula for the SARSA algorithm:

$$Q(s,a) \leftarrow Q(s,a) + \alpha[r + F(s,a,s',a') + \gamma Q(s',a') - Q(s,a)] \tag{2}$$

where $F(s,a,s')$ is the general form of the shaping reward.

Even though reward shaping has been powerful in many experiments it quickly turned out that, when used improperly, it can change the optimal policy [24]. To deal with such problems potential-based reward shaping was proposed [21] as the difference of some potential function Φ defined over a source s and a destination state s':

$$F(s,a,s',a') = \gamma\Phi(s') - \Phi(s) \tag{3}$$

where γ must be the same discount factor as used in the agent's update rule (see Equation 1).

Ng et al. [21] proved that potential-based reward shaping, defined according to Equation 3, guarantees learning a policy which is equivalent to the one learned

without reward shaping in both infinite- and finite- state MDPs. In undiscounted, finite-state MDPs for the guarantee to hold all policies must always eventually reach an absorbing state. For discounted and/or infinite-state MDPs other conditions must also be met, but all experimental work in this paper will be in undiscounted, finite-state SMDPs. Furthermore, if the transition probabilities and reward function are unknown, this becomes the only method to define any additional reward without altering the goal of the agent.

Wiewiora [35] later proved that an agent learning with potential-based reward shaping and no knowledge-based Q-table initialisation will behave identically to an agent without reward shaping when the latter agent's value function is initialised with the same heuristic knowledge represented by $\Phi(s)$. Therefore, both methods can be used with the same results expected.

To include background knowledge regarding favourable actions in reward shaping whilst maintaining the guarantees of policy invariance, further conditions must be met [9].

Specifically, Wiewiora et al. [9] identified two methods; *look-ahead advice*, formally defined in Equation 4, and *look-back advice*, formally defined in Equation 5.

$$F(s, a, s', a') = \gamma\Phi(s', a') - \Phi(s, a) \tag{4}$$

Look-ahead advice shapes an agent's reward when moving from state s to s' by action a based on the difference in potential between state-action pairs (s, a) and (s', a') where a' is defined as in the agent's update rule. Therefore, if using SARSA, a' will be the next action the agent will take or, if using Q-learning, the highest valued action in state s'. With either algorithm, at this time when using look-ahead advice action a' has not yet been performed. For look-ahead advice to maintain policy invariance the agent's policy must choose the action with the maximum sum of both Q-value and potential.

$$F(s, a, s', a') = \Phi(s', a') - \gamma^{-1}\Phi(s, a) \tag{5}$$

Alternatively, look-back advice shapes an agent's reward when moving to state s'' after action a' is used in state s' based on the difference in potential between state-action pairs (s, a) and (s', a') which have now both already occurred. Look-back advice must be used with an on-policy learning algorithm (e.g. SARSA) and a policy invariant to a constant addition to all actions in a state (e.g. ϵ-greedy).

An agent using look-back advice is not guaranteed, but is empirically demonstrated, to converge to the same Q-values as the agent would have without advice. Regardless, Wiewiora et al. [9] recommend look-ahead advice for when the prior knowledge is predominately state-based whilst look-back advice is recommended for when the prior knowledge is predominately action-based.

The term potential-based advice should not be confused with the more general term advice as used in work such as Maclin and Shavlik's advice taking agents [16].

The general term refers to giving information to agents on how to overcome certain situations whilst potential-based advice refers to a specific method of giving state-action domain knowledge to an agent. Both achieve the same result, but the latter is an implementation of the former (as too is potential-based shaping). Although potential-based advice is similar to potential-based shaping, the two differ in the domain knowledge they can represent. Ng's potential-based shaping can only represent state-based knowledge, whilst Wiewiora's potential-based advice can represent state-action-based knowledge. The convention to name these methods was set in the original paper proving how to incorporate action-based knowledge into reward shaping [9] and throughout the remainder of the paper we will stick to this also.

Typically, MAS solved by reinforcement learning require function approximation to represent the value function. Even though with function approximation the optimal policy might not be representable, the application of potential-based reward shaping and advice is still justified as previous empirical work has demonstrated the successful combined application of tile coding, a common method of function approximation [30], with both methods [11, 9].

3.1. *Multi-Agent Reward Shaping*

In this paper, we address the issues of applying potential-based reward shaping and advice in the context of multi-agent learning. Most previous work on the theoretical guarantees of potential-based reward shaping [21, 9] have been based on the assumption of a single-agent but many can be extended to provide similar guarantees when multiple agents are simultaneously learning [8].

Previous applications of potential-based reward shaping to multi-agent reinforcement learning have been implemented in relatively trivial problem domains [1, 17]. Both showed the typical beneficial result of potential-based reward shaping in single-agent reinforcement learning, decreased time to convergence, but Babes et al. [1] also witnessed changes in the probability of converging to different joint policies, an observation unique to multi-agent, potential-based reward shaping.

With multiple individual learners, applying any reward shaping alters only the reward function of that individual. The joint properties of the problem domain, the set of all states, the set of all joint actions and the transition function, are unchanged. Potential-based reward shaping adds no preference to converge to any policy. The shaped agent's best response policy to a fixed set of joint policies is, therefore, the same as had the same agent not received potential-based reward shaping. The other agents, unaffected by the shaping, will still value each policy the same and so will have in no way altered the set of possible Nash Equilibria. [8]

Similarly, the proof of Wiewiora et al. [9] regarding potential-based advice again showed that no preference was added to converge to any policy provided the specific conditions of either look-ahead or look-back advice were met. Therefore, we hypothesise these methods are also applicable to MAS with the addition of state-action based domain knowledge in this form causing no resulting modification to

the system's points of Nash Equilibrium. To the best of our knowledge, we are the first to demonstrate the application of potential-based advice to a MAS.

For both potential-based reward shaping and advice, although the same points of equilibrium still exist, whether the agents will still converge to the same one is not guaranteed. Reward shaping alters an agent's exploration. In single-agent this will only alter how long an agent takes to converge to the optimal policy as the convergence is guaranteed. However, when multiple agents are learning there are commonly multiple points of equilibrium to which the system can converge. An alteration in the exploration path of one agent may cause the collective to converge upon a different joint policy. The new joint policy may represent a Nash Equilibrium of either higher or lower global utility. [8]

In effect, given an ideal heuristic, potential-based reward shaping will increase the probability of converging towards the highest global utility Nash Equilibrium [8]. A number of other techniques have previously been developed to increase this probability. The most well-established method, Collective Intelligence (COIN) [36], encourages system designers to develop individual reward functions aligned with the global reward function. No agent is to receive a reward unless their action was beneficial to the global utility. The new reward is not potential-based and so the goals of each agent may have changed. Such an approach requires expert knowledge of the problem domain, allowing calculation of how much effect an agent's action has had on neighbouring agents. If available, this technique can be very useful with multiple published applications demonstrating its ability [33, 36]. However, this approach requires knowledge of and the ability to modify the original reward function. Meanwhile potential-based reward shaping does not and can, therefore, treat any existing agent as a black box, not modifying any existing code, simply providing an additional reward.

4. RoboCup Soccer

RoboCup is an international project[a] which aims at providing an experimental framework in which various technologies can be integrated and evaluated. The overall research challenge is to create humanoid robots which would play and win against world champion humans. Since, the full game of soccer is complex, researchers developed several simulated environments which can be used to evaluate techniques for specific sub-problems. One such sub-problem is the KeepAway[b] task [26, 27]. In this task (see Figure 1), N players (keepers) learn how to keep the ball when attacked by $N-1$ takers within a small, fixed area of the football pitch.

This task is multi-agent [37] in its nature, with elements of both cooperation and competition. Overall, there are three types of high level behaviour in this task. First let us consider the agents trying to maintain possession of the ball; the keepers.

[a]See http://www.robocup.org/ for more information
[b]See http://userweb.cs.utexas.edu/∼AustinVilla/sim/Keepaway/ for more information

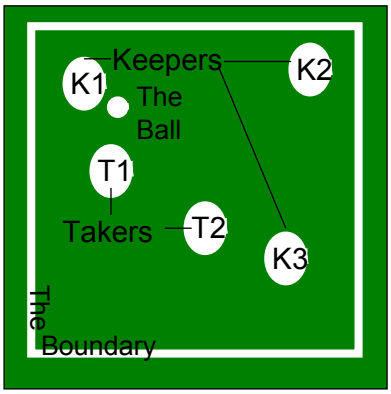


Fig. 1: Snapshot of a 3 vs. 2 KeepAway game.

For keepers there are two distinct situations, either the keeper has possession of the ball or it does not. If not in possession of the ball, a keeper needs to move to a position convenient to receive the ball from the keeper that does have possession. The second behaviour is that of the keeper in possession of the ball, who must decide which other keeper to pass to or whether to maintain possession and wait for an appropriate time to pass.

The third and final behaviour is, that of the opposing team of agents trying to win possession of the ball; the takers. The takers must decide whether to close down the keeper in possession of the ball and attempt a tackle or to instead mark one of the keepers off-the-ball and attempt to intercept an incoming pass from the keeper on-the-ball.

4.1. *Multi-Agent Learning in RoboCup Soccer*

Previous work has attempted to learn the keepers' behaviour whilst in possession of the ball using reinforcement learning whilst the takers and keepers off-the-ball (i.e. not in possession of it) adhere to a hand-coded policy [27, 7].Although this existing work has multiple agents learning during each episode, the implementation is not a true multi-agent learning example. At any time only one agent is learning, namely the keeper in possession of the ball, and all other agents are following fixed hand-coded policies. Therefore, the agent is learning within a static environment.

To make the problem of learning the keepers' behaviour multi-agent, both the behaviour of the keeper with the ball and the keepers without the ball can be updated simultaneously. This has been previously studied with a combined temporal difference and policy search solution [15]. Alternatively, learning just the behaviour of keepers without the ball would also be a multi-agent learning problem, provided games of 3v2 or more players, as at all times at least two keepers would be off-the-ball.

Another multi-agent learning task possible using the KeepAway simulator is learning the behaviour of the takers. This task has previously [19], and will be throughout this paper, referred to as *TakeAway* to differentiate between experiments learning the takers' behaviour and those learning the keepers' behaviour. When learning the behaviour of the takers, the behaviour of the keepers is fixed to a hand-coded policy first introduced by Stone et al.[27].

Previous attempts to learn the behaviour of takers proved relatively successful [14, 19] and were a useful resource when attempting to develop novel approaches. The first basic learning taker was developed by Iscen and Erogul [14] using SARSA reinforcement learning with tile coding to decide the action of a taker every 15 cycles. This work emphasised that allowing a taker to decide an action on every cycle caused indecisiveness in the agent because the short time elapsed between decisions did not allow adequate time for the true benefit or cost of an action to be realised. In experiments allowing decisions to be made every cycle takers oscillate between decisions causing poor performance. This observation was later witnessed again by Min et al. [19].

There still remains large room for improvement in the development of a learning taker as the more challenging a taker can become, the more it will challenge researchers interested in learning the behaviours of keepers. The work we have undertaken has resulted in takers performing significantly better than the performances reported in both these papers [14, 19] against the same opposing keepers in games with the same set up and in games more challenging to the takers.

As they contain elements of both competition and cooperation, these problem domains provide a suitable test bed for more generally applicable research into multi-agent reinforcement learning. With the learning keepers, we have used potential-based reward shaping with state-based domain knowledge to complete the first empirical demonstration in a complex MAS. Then, given the learning takers, we have further developed this demonstration and present the first demonstration of potential-based advice in a MAS.

5. KeepAway

5.1. *Proposed Method*

In this section we provide more details on our learning keepers and the reward shaping techniques used. In our investigation we compare the performance of reinforcement learning keepers without reward shaping (the baseline learner) to keepers using reward shaping.

5.1.1. *Baseline Learner*

Our baseline learning keeper learns the GetOpen behaviour of keepers not in possession of the ball, originally introduced by Kalyanakrishnan and Stone [15]. Keepers in the original work [15] learn two separate policies; one for whilst in possession of

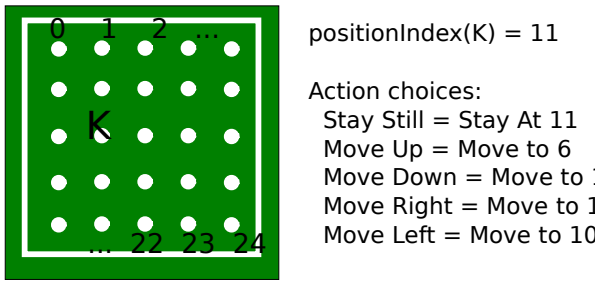


Fig. 2: The 25 Possible Locations of Keepers when Off-The-Ball and 5 Example Actions Given a Keeper at K.

the ball and one for while not. Ours will only learn the policy of keepers off-the-ball. This limitation was necessary because, as suggested in the original work [15], our early experimental testing showed that learning both the behaviours of keepers with the ball and without is infeasible with temporal difference solutions to both.

To overcome the difficulty of learning both policies by temporal difference, the existing work [15] updates a single off-the-ball policy with experiences from all agents. Each agent would then make action decisions by a policy search method using the globally available network. Alternatively, to later extend the baseline keeper with potential-based reward shaping our implementation must use temporal difference algorithms. Therefore, the keepers will each learn their own off-the-ball policy by temporal difference.

As justified earlier, learning the GetOpen policy alone is a multi-agent learning problem. Therefore, learning the off-the-ball behaviour alone provides us with a learning problem that is both multi-agent and achievable with temporal difference methods.

When the keeper is in possession of the ball, a fixed hand-coded behaviour originally defined by Stone et al. [27] will be followed. Meanwhile, whilst not in possession of the ball, the keeper must choose to move up, down, left, right or stay still based on the two dimensional pitch being divided into 25 equidistant points as illustrated in Figure 2. To learn when to perform these actions we use the SARSA algorithm with tile coding and ϵ-greedy action selection method, as in the original work on learning keepers in KeepAway [27].

After each completed action the agent is rewarded according to how much time has elapsed since the action began. This way the keepers are encouraged to maximise the time they, as a team, maintain possession. It is important in these experiments to reward proportional to the time taken, as the actions in both KeepAway and TakeAway take differing lengths of time to complete. In effect, the true model of both problem domains is a Semi-Markov Decision Problem. If the agents were instead rewarded proportional to the number of completed actions, the team would instead learn to perform lots of short actions and so would not learn the desired

behaviour of keeping or winning possession.

Keeper	GetOpen
$dist(K_1, K_2)$	$dist(K_1, K_2)$
$dist(K_1, K_3)$	$dist(K_1, K_3)$
$dist(K_1, T_1)$	$dist(K_1, T_1)$
$dist(K_1, T_2)$	$dist(K_1, T_2)$
$min_{j \in 1,2} dist(K_2, T_j)$	$min_{j \in 1,2} dist(K_2, T_j)$
$min_{j \in 1,2} ang(K_2, K_1, T_j)$	$min_{j \in 1,2} ang(K_2, K_1, T_j)$
$min_{j \in 1,2} dist(K_3, T_j)$	$min_{j \in 1,2} dist(K_3, T_j)$
$min_{j \in 1,2} ang(K_3, K_1, T_j)$	$min_{j \in 1,2} ang(K_3, K_1, T_j)$
$dist(K_1, C)$	$dist(K_1, K)$
$dist(K_2, C)$	$min_{i,j \in \{2,3\} X \{1,2\}} ang(K_i, K_1, T_j)$
$dist(K_3, C)$	
$dist(T_1, C)$	
$dist(T_2, C)$	
$positionIndex(K)$	$positionIndex(K)$

Table 1: State Representations for Learning Keepers

To increase the number of learning problems evaluated with potential-based reward shaping and to demonstrate more of the theorised outcomes of applying reward shaping in MAS, two different state representations were implemented. Both state representations are documented in Table 1. To clarify, K is the agent itself, K_i is the i-th closest keeper to the ball, T_j is the j-th closest taker to the ball and C is the centre of the pitch. The method $ang(x, y, z)$ returns the angle with vertex y and edges yx and yz, $dist(x, y)$ returns the distance between x and y, $min_{j \in 1,2}$ returns 1 or 2 dependent on which returns the smallest value when input to the next method and $positionIndex(K)$ returns the index of the point the agent is closest to out of the 25 equally distributed points keepers can move to.

The keeper state representation is based on the early work by Stone et al. [27] and the GetOpen state representation is similar to the approach of Kalyanakrishnan and Stone [15].

Moving to each agent learning a separate off-the-ball policy from a joint policy search method learning a single shared off-the-ball policy for all keepers, as implemented in the existing work [15], is expected to produce lower average performance in our baseline learning keepers compared to those previously produced [15]. However, the performance of the baseline learner will be sufficient to show that potential-based reward shaping can be applied to an existing reinforcement

learning solution to reduce training time and possibly improve final performance without modification of the original learning agent.

5.1.2. *Separation-Based Reward Shaping*

The separation-based reward shaping function is our first attempt to apply potential-based reward shaping to a complex, MAS. This agent is intended to show that the use of reward shaping is both applicable and beneficial in multi-agent reinforcement learning.

Specifically, the domain knowledge we have applied states that keepers can improve their performance by spreading out. By following this principle, each keeper off-the-ball creates a unique angle for the keeper with the ball to pass along. Therefore, one taker cannot mark multiple keepers at once as they could if the keepers stuck together.

We have implemented a reward shaping function that encourages separation by adding the change in distance between the keepers to the reward they receive from the basic learning algorithm. Based on Equation 3, this shaping function can be formalised as:

$$\begin{aligned} DistanceBetweenKeepers(s) = \quad & DistanceBetween(K_1, K_2) \\ & + DistanceBetween(K_1, K_3) \end{aligned} \tag{6}$$

$$\begin{aligned} F(s, s') = \gamma & DistanceBetweenKeepers(s') \\ & - DistanceBetweenKeepers(s) \end{aligned} \tag{7}$$

Assuming our domain knowledge is beneficial, we hypothesise the addition of this potential-based function will increase the agents' final performance. Alternatively, if the baseline learners do converge to a policy of equivalent performance, the shaped agents will know from the beginning to attempt to separate and so will converge quicker.

5.2. *Experimental Design*

The experiments undergone were performed in RoboCup Soccer Simulator v11.1.0 compiled against RoboCup Soccer Simulator Base Code v11.1.0. The KeepAway player code used was keepaway-player v0.6. Takers were based upon the hand-coded policy publicly available in this release and keepers were implemented by adding to the provided keeper our own code for reinforcement learning and reward shaping.

For keepers both with and without reward shaping, the SARSA algorithm of reinforcement learning was used with the parameters; $\alpha = 0.125, \gamma = 1.0$ and $\epsilon = 0.01$. For function approximation a tile coding function with 14 or 11 groups (dependent on Keeper or GetOpen state representation respectively) of 32 single-dimension tilings was used. All keepers used one group per feature in the state representa-

tion. Angles were divided into ten degree intervals and distances into three meter intervals. Position indices were not approximated.

The base reward function, used by all agents, is a positive reward equal to the time passed between action choices with a large negative reward (-50) upon the start of a new episode to punish the receivers for losing possession. The supplemental reward from the shaping functions must be scaled to interact appropriately with this. A poor matching of scaling to the base reward function and state representation can reduce the gain in performance of a good heuristic. For these experiments, the value of separation was doubled before it was added to the basic reward function of the agents when receiving reward shaping. This scaling factor was found through experimental testing. Therefore, it may not be the optimal setting. However, it is sufficient to show the improvement in performance the methods are capable of.

Experiments were performed on pitches of sizes 20×20 meters. These values were chosen due to the complexity of the learning task. We will explore scaling up a problem domain when we look at the simpler problem of TakeAway.

Experiments were repeated at least 25 times. The results provided in Section 5.3 illustrate the change in average episode length over all repeated experiments against time. Given that we are learning the behaviour of the keepers at this time, we are aiming to maximise the length of the average episode.

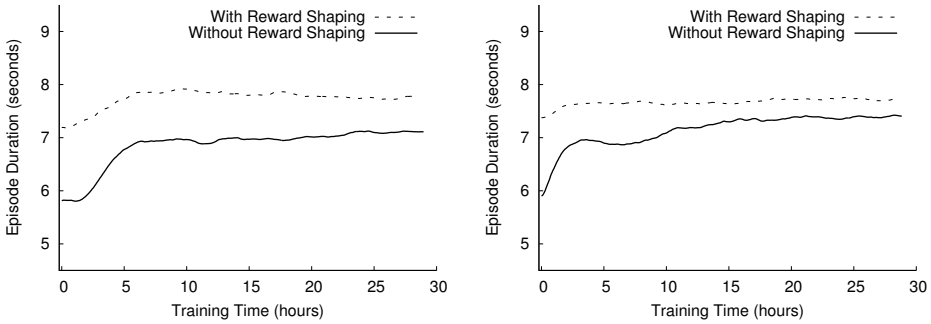
5.3. *Results*

The results of experiments comparing our baseline learner with those also receiving potential-based reward shaping are supportive of the use of this method in MAS. Illustrated in Figure 3, keepers learning regardless of state representation were improved by using separation-based shaping.

In these experiments the baseline learner was treated as a black box with the only modification required being to send each agent an additional reward based on the potential of the states it has transitioned from and to. This is a useful observation because it demonstrates how any existing MAS with a reinforcement learning solution could, with some domain knowledge, benefit from adding potential-based reward shaping.

Furthermore, both hypothesised benefits of multi-agent, potential-based reward shaping have occurred. Figure 3a shows shaped agents learning a significantly better ($p = 1 \times 10^{-8}$) performing joint policy than the baseline keepers, but, taking approximately the same time to do so. This result empirically demonstrates that if a group of agents' exploration is modified sufficiently a different joint policy can be reached. If the heuristic is suitable, as in this example, the shaped agents are encouraged not to choose sub-optimal actions and so the different joint policy found is representative of a better performance. However, as we will show in later experiments, the ability to converge to different equilibria can negatively affect final performance if a misleading heuristic is used.

In Figure 3b, we have demonstrated that agents learning with potential-based



(a) Keeper State Representation (b) GetOpen State Representation

Fig. 3: 3 Learning Keepers vs. 2 Hand-Coded Takers.

reward shaping can reach a joint policy, which does not improve significantly with further learning, quicker than an agent without domain knowledge. The shaped agents can learn quickly because their exploration is directed. Furthermore, the joint policy learnt is still significantly better than the best learnt by the baseline keeper ($p = 0.07$).

As the same heuristic, separation-based shaping, has been used twice in the same problem domain and was beneficial both times we know the heuristic is suitable for discouraging sub-optimal actions. However, how the heuristic is scaled in comparison to and matches with the other settings of the agent will affect how much benefit is gained.

To conclude, the experiments on KeepAway have successfully demonstrated that reward shaping can be beneficial in complex MAS. To further add to this body of work, more experiments are required to support the claim that these methods are generally applicable. In the next section we will discuss results for a distinct learning problem in the same environment. Learning the takers' behaviour occurs in the same simulator. However, it is a significantly different task with opposing goals to the behaviour learnt in these experiments.

6. TakeAway

6.1. *Proposed Method*

In this section we provide more details on our learning takers and the reward shaping techniques used. In our investigation we compare the performance of reinforcement learning takers without reward shaping (the baseline learner) to takers using one of three types of reward shaping, detailed below, including the first published applications of potential-based advice in MAS.

6.1.1. *Baseline Learner*

Our baseline learning taker combines the work of both previous papers [14, 19] on learning takers in KeepAway. As in both these papers, the takers can on each update choose either to tackle the keeper with the ball or mark a specific keeper. To tackle, the taker chases the ball and attempts to gain possession of the ball. To mark a keeper, the taker moves close to the keeper positioning itself between the ball and the keeper so as to gain possession if the ball is passed to that keeper.

To learn when to perform these actions we use the SARSA algorithm with tile coding and ϵ-greedy action selection method, as Iscen and Erogul [14] did. Then we use the state representation and reward function, -1 for every cycle the episode continues to run and +10 for ending the episode, designed by Min et al. [19]. Given the observations made by both papers we update the policy and make new action choices only after every 15 cycles.

Image Label	Formal Definition
a	$dist(K_1, K_2)$
b	$dist(K_1, K_3)$
c	$dist(K_1, T_1)$
d	$dist(K_1, T_2)$
e	$dist(K_1, C)$
f	$dist(K_2, C)$
g	$dist(K_3, C)$
h	$dist(T_1, C)$
i	$dist(T_2, C)$
j	$min_{j \in 1,2} dist(K_{2-mid}, T_j)$
k	$min_{j \in 1,2} dist(K_{3-mid}, T_j)$
l	$min_{j \in 1,2} ang(K_2, K_1, T_j)$
m	$min_{j \in 1,2} ang(K_3, K_1, T_j)$

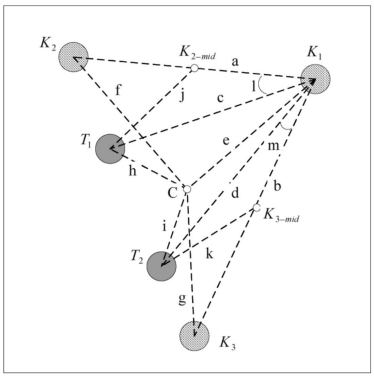


Fig. 4: State Representation for Learning Takers [19].

Figure 4 documents the features received by the takers in the chosen state representation. All reoccurring methods and symbols in the taker state representation represent the same meaning as previously introduced in the state representation descriptions of the baseline learning keepers. The one new symbol, K_{i-mid}, marks the mid-point between the keeper closest to the ball and the i-th closest keeper to the ball.

We chose the state representation from Min et al. [19] as opposed to the one used by Iscen and Erogul [14], because the latter represented fewer observations of the environment and we expected this to limit the performance of learning takers. Our experiments presented later show that the more detailed state representation improves the performance of the basic learning taker supporting our expectations

and providing a useful comparison to the more novel approaches we discuss in the following two subsections.

6.1.2. *Separation-Based Reward Shaping*

As with the keepers learning with reward shaping the learning takers can also benefit from increasing their separation. By following this principle, they are able to limit the passing options of the keepers and reduce the time the keepers maintain possession.

This agent is intended to again show the benefits of applying potential-based reward shaping to agents learning in MAS and to support the argument that the method is generally applicable to MAS as a whole.

6.1.3. *Role-Based Advice*

In experiments with the previous agent based upon a separation-based reward shaping, all taker agents will be homogeneous. A more interesting problem is that of heterogeneous agents, whereby different agents cooperating on the same team combine different skills to outperform their homogeneous counterparts [2].

Given the previous hypothesis, that takers sticking together is detrimental to performance, more complex prior domain knowledge can be incorporated stating that it is beneficial for one taker to tackle and another to fall back and mark.

In effect, this new domain knowledge defines two roles; one of a tackling taker and one of a marking taker. We thus use reward shaping to encourage these heterogeneous roles in the learning takers. By rewarding one taker for choosing to tackle when previously choosing a marking action and punishing it when it changes from choosing tackling to now marking the agent will be encouraged to tackle. Please see Listing 1 for clarification, the specific values of Reward and Punish will be defined shortly. A similar approach reversing the punishment and reward will then encourage the other taker to mark.

Listing 1: Tackler Heterogeneous Role Shaping Function

```
if not(a == a')
then if (a' == TacklingAction)
  then  F(s,a,s',a') = Reward
  else  F(s,a,s',a') = Punish
else F(s,a,s',a') = 0
```

As this domain knowledge is action-based it becomes an implementation of potential-based advice [9] and not simply potential-based reward shaping. Therefore, additional requirements must be met if the addition of no preference to any one policy is to remain. As the knowledge we are implementing is solely action-based we will use look-back advice as recommended by [9]. Look back advice requires an on-policy learning algorithm and action selection based on relative differences in

value, not absolute magnitude. Both of these conditions have been met by design of the baseline agent by the SARSA algorithm and ϵ-greedy policy. The shaping function of look back advice is:

$$F(s, a, s', a') = \Phi(s', a') - \gamma^{-1}\Phi(s, a) \tag{8}$$

where γ is the same discount factor as the environment and $\Phi(s, a)$ is the potential of state-action pair (s, a).

Therefore, when considering the taker assuming the role of tackler, we assign any state-action pair with a tackling action a potential of 2 and any state-action pair with a marking action a potential of 1. Combining these potentials and Equation 8, Table 2 lists the additional rewards received by tackling agents. The value given for moving from a marking action to a tackling action is the Reward value referred to in Listing 1 and the value for moving from a tackling action to a marking action is the Punish value.

These roles are not hard-code. We are not limiting the action choices available to the takers. Both takers can still choose either to mark or tackle and reinforcement learning will still have them explore the use of both action choices. Therefore, when it is necessary for the marking agent to tackle he will still make the correct decision and tackle, but in general it will choose to mark as the reward shaping function applied will make this appear more lucrative.

It is hoped that these two roles are beneficial. Assuming they are and the heuristic is suitably matched to the original, unshaped reward function and state representation, we hypothesise the agents will converge quickly to an equal or better performance than the baseline learner. If correct, the successful application of this potential function will demonstrate both the applicability of potential-based advice in MAS and a use of the method to encourage heterogeneous roles.

a	a'	Formula	Value
Mark	Tackle	$2 - \gamma^{-1}$	$+1$
Tackle	Mark	$1 - 2\gamma^{-1}$	-1

Table 2: Shaping Values of a Tackling Taker given $\gamma = 1$

6.1.4. *Combining Shaping Functions*

Finally, we have also considered the incorporation of both pieces of domain knowledge into one team of takers. This way the takers can be encouraged to take roles but also consider the benefit of separating.

When combining shaping functions it is important that each is scaled individually because to calculate the potential difference of both states and scale the sum

would give a different meaning to the resulting reward shaping, it would not accurately represent the domain knowledge intended. Therefore, the potential-based reward shaping function changes from Equation 3, given in Section 3, to:

$$F(s, a, s', a') = \tau_1 F_1(s, a, s', a') + \tau_2 F_2(s, a, s', a') \tag{9}$$

where F_1 and F_2 are the shaping functions for role-based advice and separation-based shaping respectively and τ_1 and τ_2 are two separate scaling factors.

In early empirical tests, scaling variables set to emphasise the role-based advice function showed the best performance. Therefore, the combined shaping agent will also emphasise the role-based advice function. This agent will still include the separation-based reward shaping function but by scaling the function appropriately it will have less of an impact on the resulting behaviour than the encouragement to take up a specific role.

It is expected that as this agent will benefit from both pieces of domain knowledge that this will be our best performing agent and as such will be a beneficial contribution to the RoboCup KeepAway research field.

6.2. *Experimental Design*

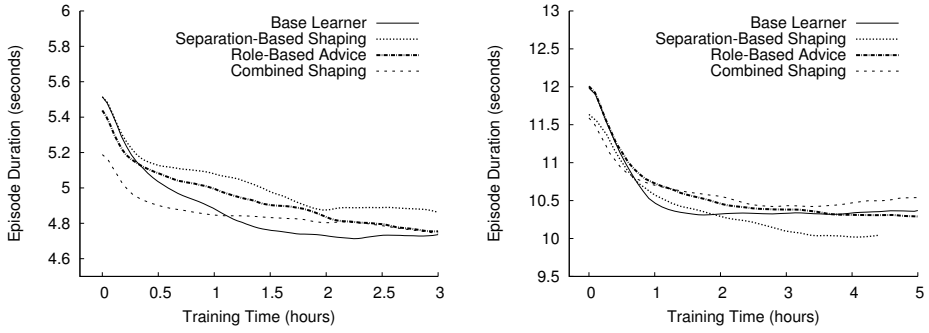
The experiments undergone were again performed in RoboCup Soccer Simulator v11.1.0 compiled against RoboCup Soccer Simulator Base Code v11.1.0. The KeepAway player code used was keepaway-player v0.6. This time the keepers were based upon the hand-coded policy publicly available in this release and takers were implemented by adding to the provided taker our own code for reinforcement learning and reward shaping.

For takers both with and without reward shaping the SARSA algorithm of reinforcement learning was used with the parameters; $\alpha = 0.125, \gamma = 1.0$ and $\epsilon = 0.01$. For function approximation a tile coding function with 13 groups of 32 single-dimension tilings was used. All takers used one group per each feature in the observation and split angles into ten degree intervals and distances into three meter intervals.

For the separation-based reward shaping agent the value of separation was doubled before added to the basic reward function, and for the role-based advice approach agents were scaled by 5. Given that $\gamma = 1.0$ this effectively means agents with role-based advice are either rewarded or penalised by 5 for changing their action from marking to tackling and vice versa.

When combining shaping functions we wanted to emphasise the heterogeneous role knowledge and so for changing their action these takers were either rewarded or penalised by 10 and for separation the change in distances were simply added. Given that role-based advice is to be the first shaping function in the combination, formalised in Equation 9, this corresponds to a τ_1 of 10 and a τ_2 of 1.

Experiments were performed on pitches of sizes 20×20, 30×30, 40×40, and 50×50 meters. These values were chosen to show the performance of our takers in



(a) 20x20 (b) 30x30

Fig. 5: 2 Learning Takers vs. 3 Hand-Coded Keepers.

similar contexts to previous work on learning the behaviour of takers and also in more complex problem domains.

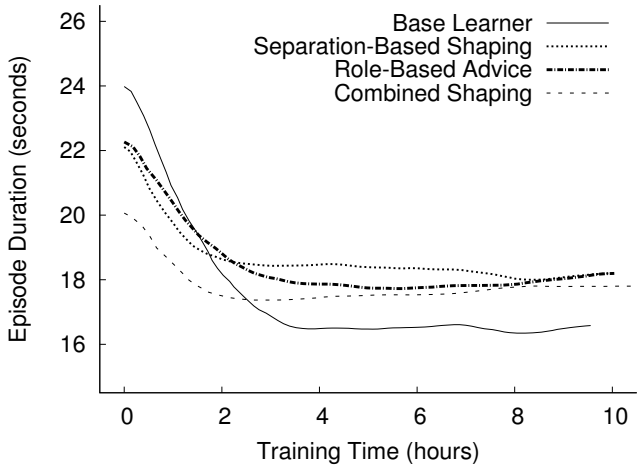
Experiments with each combination of pitch size and reward shaping function were repeated at least 25 times. The results provided in Sections 6.3 and 6.4 illustrate the change in average episode length over all repeated experiments against time. Given that we are now learning the behaviour of the takers, we are instead now aiming to minimise the length of the average episode.

6.3. *Results*

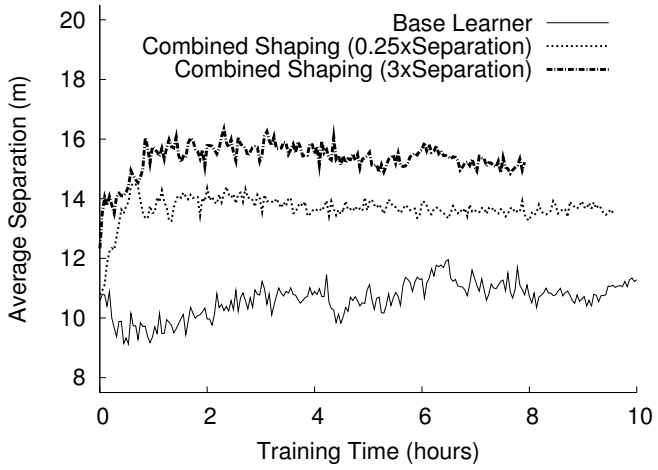
In experiments on the simplest domains, all agents learnt good policies quickly with no significant difference ($p > 0.2$) in performance. For both pitches of size 20x20 and 30x30, illustrated in Figures 5a and 5b, it is important to consider that both axes represent small changes in time in their given dimension and the differences between agents is both brief and insignificantly small (only 0.4 seconds for pitch size 20x20). Therefore, TakeAway at pitches of this size is too simple to gain much benefit from reward shaping.

In problem domains where reinforcement learning alone can quickly learn a policy of good performance, the additional work of designing a heuristic and implementing reward shaping, however simple that may be, is unnecessary. These methods are more beneficial in complex problem domains where reinforcement learning alone takes a long time to converge and has a large difference in performance between the initial policy and the final policy converged to.

These results, however, have been included for comparison to previous work on learning takers. Our baseline agent is approximately equivalent to the best performing, learning taker from the existing published attempts [19] which is quoted as converging on average to win possession in 5.8 seconds in games of 3v2 on pitches of size 20x20. All learning takers, both the existing and our own baseline learner,



(a) Overall Performance



(b) Average Separation

Fig. 6: 2 Learning Takers vs. 3 Hand-Coded Keepers at 40x40.

outperform the standard hand-coded takers defined by [27] that perform consistently around 15 seconds. Therefore, the baseline learner we have developed is both a suitable and highly competitive test agent to compare our approaches to.

At a pitch size of 40x40 the problem appears to become sufficiently difficult, with the baseline learner unable to converge quickly as seen in Figure 6a. With this level of difficulty a clear difference in agents is now evident. All shaped agents immediately benefit from the additional domain knowledge with statistically significant differences ($p < 0.05$) in initial performance to the baseline takers. In particular,

the combined shaping agent is highly significantly better ($p = 1.6 \times 10^{-5}$).

During the early episodes of training, all shaped agents improve performance at a visually similar rate to the baseline learner and so maintain their positive difference in performance. After an hour of training the learning of takers using reward shaping begins to slow and the average performance of the baseline learner starts to catch up. At two hours of training, the performance of all agents is equivalent ($p = 0.9$) but at 8 hours the baseline learner significantly outperforms the shaped agents ($p < 0.005$).

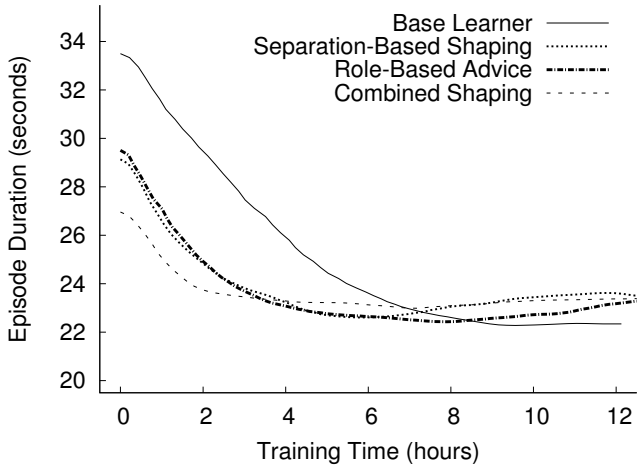
Figure 6b[c] shows that increasing the scale of the separation shaping function causes agents to separate further on average. This is empirical evidence that, as has been proven theoretically [8], agents receiving reward shaping may learn different joint policies when in a common environment.

Convergence to a different joint-policy, at 40x40, has caused the difference in performance between agents with shaping and the baseline learner. The heuristics used are poorly matched to the other settings of variables at 40x40. The agents still benefit from directed exploration, by initially improving performance quicker than the baseline learner, but suffer as their final policy is different and represents a behaviour of lower performance. It would be an implementation decision to prioritise either the reduced training time of the shaped agents or the higher final performance of the baseline takers.

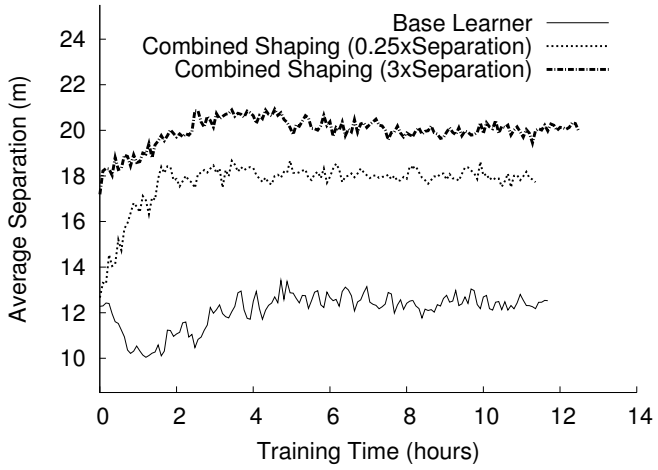
Furthermore, it can be observed that the shaped agents after quickly achieving a good performance degrade over time. This was also observed previously by Wiewiora et al. [9] when using a negative base reward function. When the Q-values at convergence were negative and exploration had been directed, in this case by potential-based advice, although the optimal policy was found all other policies appeared better as their initial values were higher than the true values of the good policy [9]. The same has occurred here in a multi-agent context. Our base reward function is also negative, therefore, by initialising our Q-value to zero we have given an optimistic value to all states and actions. Once a good equilibria is found, replacing the optimal policy in the single-agent example, the agent values it less than all other policies and so begins to explore these instead. Given sufficient time the agents would begin to recover from this, as will be seen later in Figure 8b.

A better solution could be to initialise the Q-table pessimistically by setting all values in the beginning to less than the lowest value in the Q-table of an agent that has already converged. For example, given that all takers in experiments with all combinations of number of players and pitch sizes can win possession after convergence in less than 30 seconds a pessimistic initialisation may be to start with all

[c]The two combined agents documented in this figure and Figure 7b represent the best tuned solutions found for 40x40 and 50x50. It is worthwhile to note that changes in environment parameters will often require a change in scaling parameters when combining reward shaping functions. In this example, the change in pitch size requires a scaling of 0.25x at 50x50 and a scaling of 3x at 40x40 for optimal performance in each context.



(a) Overall Performance



(b) Average Separation

Fig. 7: 2 Learning Takers vs. 3 Hand-Coded Keepers at 50x50.

state-action pairs valued to -30 instead of simply 0. With this initial value, upon finding a good policy of less than 30 seconds the agents would prefer this policy and the period of deteriorating performance should not occur.

The results on pitches of 50x50, as illustrated by Figure 7, show a problem domain more suitable to the use of reward shaping. As previously seen in the change from pitch sizes of 30x30 to 40x40, there is a significant rise in difficulty when increasing the pitch size from 40x40 to 50x50. Given the yet again higher difficulty, a more significant improvement can and has been witnessed.

Firstly, there is now a highly significant difference ($p < 4 \times 10^{-8}$) between the initial performance of all shaped takers and the baseline learner. The most significant being between the baseline and takers receiving the combined advice of both heuristics ($p = 2 \times 10^{-17}$).

This gain in performance remains roughly constant throughout the first 4 hours of training. It then begins to shrink but still outperforms the baseline learner for up to approximately 8 hours. Even after the first 8 hours of training, the baseline learner can only match the performance of the novel approaches and never significantly outperforms any of them ($p > 0.1$ after 11 hours).

Finally, the agents solely encouraged to take heterogeneous roles did adhere to the encouragement and after convergence were seen to almost exclusively stick to their assigned roles. They did not, however, follow their assigned roles blindly and did deviate occasionally from them in states where they learnt it to be beneficial. By using reinforcement learning with potential-based advice to encourage roles, these deviations from the encouraged role were possible whereas an agent with enforced roles would not provide such flexibility.

6.4. *Scaling Up*

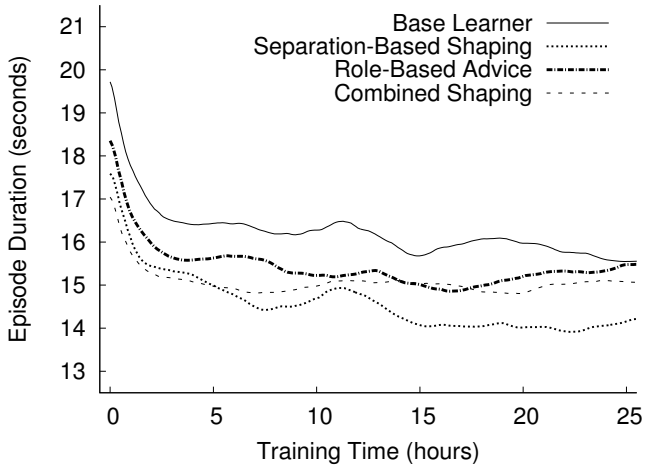
To further challenge the learning takers, more agents can be deployed. By adding agents to the learning team, cooperation becomes harder. However, to maintain the games dynamics we must also add to the keepers and so now we shall look at games of three takers versus four keepers (3v4) and four takers versus five keepers (4v5).

The experiments illustrated in Figures 8 and 9 are again the results of at least 25 repeats.

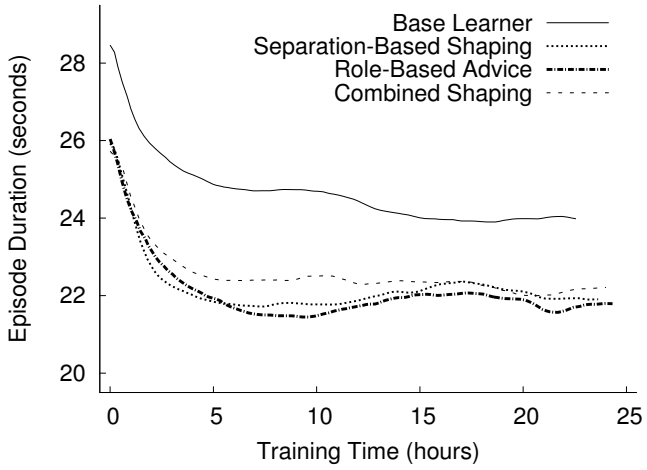
The first results of 3v4 at 40x40, illustrated in Figure 8a, a classic example of potential-based shaping/advice altering exploration sufficiently to both benefit and subtract from final performance. In this specific problem domain, the results conclude that separation-based shaping is a more suitable heuristic than role-based advice. This is apparent because the separation-based shaped agents' joint policy represents the most significantly better performance than the baseline taker ($p = 3 \times 10^{-8}$). However, both the role-based advised agents and the combined shaping agents still also manage to learn a significantly better policy than the baseline learner ($p = 4 \times 10^{-4}$ and $p = 7 \times 10^{-3}$ respectively).

At pitch sizes of 50x50, illustrated in Figure 8b, all shaped/advised agents significantly ($p < 0.03$) outperform the baseline agent in 3v4. Agents receiving separation-based shaping are again the best solution for 3v4, as they learn the policy on average two training hours quicker than the nearest competitor.

Additionally, as alluded to earlier, Figure 8b shows a team of agents recovering from the period of deteriorating performance caused by optimistic Q-value initialisation. On pitches of 50x50 in games of 3v4, the separation-based agents reach their best performance at 6 hours, then for the next 10 hours deteriorate before again beginning to improve their performance. The experiments were not long enough



(a) Overall Performance at 40x40

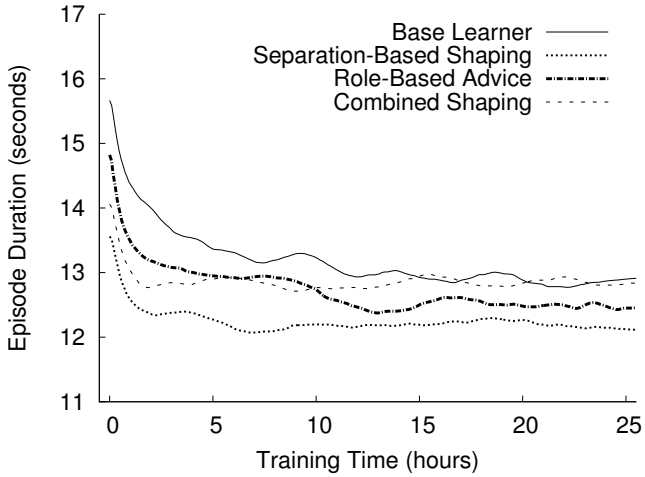


(b) Overall Performance at 50x50

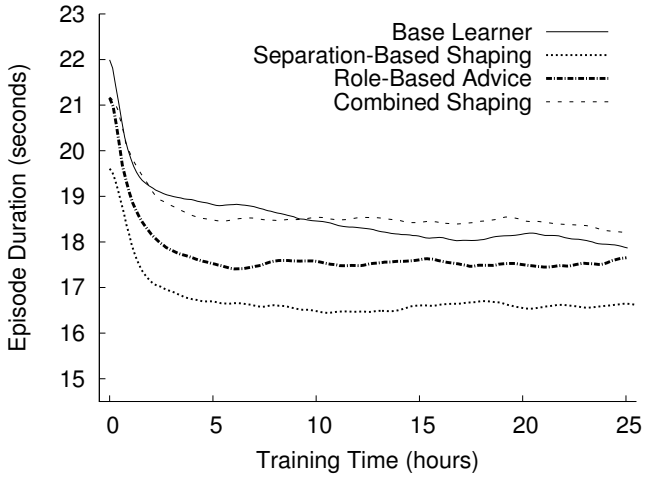
Fig. 8: 3 Learning Takers vs. 4 Hand-Coded Keepers.

to show a complete recovery to their best joint policy. Although this demonstrates that agents can eventually stop the decline in performance given sufficient time, the authors would still recommend a pessimistic initialisation as the best solution to prevent this occurring.

Increasing the number of agents again to 4v5 begins to reach the limit of scaling the learning task with regard to number of agents at these given pitch sizes. This is apparent as the performances achieved by the takers are now significantly better than at either 3v4 or 2v3. However, some conclusions can still be made and further



(a) Overall Performance at 40x40



(b) Overall Performance at 50x50

Fig. 9: 4 Learning Takers vs. 5 Hand-Coded Keepers.

evidence of the general applicability of both potential-based shaping and advice to MAS is provided by these examples.

At 40x40, illustrated in Figure 9a, all advised agents, both role-based and combined-shaped, learn joint policies equivalent of the baseline agent ($p > 0.1$) but do so quicker due to directed exploration. Again, for this problem domain the separation-based shaped agents are the superior solution as they also learn quicker by directed exploration but also learn a joint policy representative of a performance significantly better than all other agents ($p = 0.007$).

At 50x50, illustrated in Figure 9b, the difference is further exaggerated and separation-based shaping is yet again clearly the dominant method, learning the quickest and to a highly statistically significant better performance than any other team of takers ($p = 3 \times 10^{-5}$). The combined shaped takers again match ($p = 0.5$) the performance of the baseline learner and do so with less training time, showing they maintain some benefits in this problem domain. Meanwhile, the role-based heuristic regains some suitability to this problem domain by slightly outperforming the baseline takers ($p = 0.09$).

Overall, the results from increasing the number of agents have shown a better ability to scale for the separation-based shaping than the role-based advice or combined shaping. However, this is believed to be a feature of the particular heuristics and not potential-based shaping compared to potential-based advice. The reason being that the roles used were designed for teams of two. With two takers, one tackler and one marker is intuitive, however with three takers, one tackler and two markers is only intuitive if each marker sticks to a given keeper for a period of time. As it was coded the takers were only encouraged to pick a marking action and changes between which marking action were not considered. Therefore, marking takers may oscillate between marking one agent and another frequently making it harder to coordinate and subsequently breaking the benefit of the roles. This also detrimentally affected the combined shaping agents, who's exploration was modified by both heuristics, unfortunately with the role-based advice commonly having a larger effect than the more beneficial knowledge of separation-based shaping.

7. Conclusion

In conclusion, we have demonstrated the applicability and benefits of using potential-based reward shaping and advice in MAS. By incorporating domain knowledge in an agent's design the agent can, provided a good heuristic, learn in less time a joint policy representative of equal or better performance than agents learning by reinforcement alone.

Unlike previous publications, we have considered the theoretical implications [8] of using these methods in multi-agent reinforcement learning as opposed to single-agent problem domains where their proofs were originally intended to hold. Potential-based reward shaping in MAS can lead to convergence on different joint policies then had the agents learnt without the additional rewards. This, as we have demonstrated in two different learning tasks and a wide range of settings, can be beneficial by increasing final performance and/or decreasing training time required but also potentially detrimental if using a misleading heuristic or poorly matching a good heuristic to other settings.

The work here has been based entirely in fully observable problem domains, which some may consider uncharacteristic of MAS. However, by shaping/advising agents based on the potential of observations (as opposed to fully observed states) the same arguments and proofs as previously theorised presuming full observabil-

ity [8] can be used to show similar theoretical expectations in partially observable problem domains. Namely, the Nash Equilibria of a partially observable problem domain would remain the same but the agents exploration will alter and so convergence may be to a different point of equilibrium or, given an unsuitable heuristic, may not converge at all.

Although the specific reward shaping functions implemented have used domain specific knowledge the types of domain knowledge represented are generally applicable. The knowledge that keepers and takers should try to stay separate is an example of knowledge regarding how agents should maintain states relative to each other. Maintaining a state relative to either team-mates or opponents is a common type of knowledge applicable in many MAS. For example, it has been shown in the predator/prey problem domain that it is beneficial for predators to consider the relative location of its supporting predator to aid coordination [31]. Similarly, having one tackler and one marker is specific to takers in TakeAway but the knowledge that agents should specialise into roles is common in MAS. For example, again in the predator/prey problem domain, it has been shown that it is beneficial to have one predator take a hunting role and another take a scouting role [31]. Therefore, the use of potential-based reward shaping and advice could be applied in general to any MAS that would benefit from agents having these types of knowledge with the expected benefits being similar to those documented in the KeepAway and TakeAway problem domains. By empirically demonstrating potential-based reward shaping in two distinctly different learning tasks we have provided strong supporting evidence that these results will occur when the methods are added to any existing reinforcement learning solution of a complex MAS.

Furthermore, neither type of knowledge used for reward shaping in our experiments explicitly defines the solutions. Each agent's policy is still learnt by the agent, the knowledge only directs the path exploration takes. Therefore, agents are still free to explore and converge upon any equilibrium via self-learning without being limited to a pre-defined solution.

To summarise, this paper has contributed the first empirical demonstration of potential-based reward shaping in a complex MAS and the first of potential-based advice in any multi-agent problem domain. We have discussed the impact of applying such reward shaping functions when multiple individual learners are acting in a common environment and illustrated the theoretical expectations with ample examples from experimentation in the RoboCup KeepAway problem domain.

The points of equilibrium that multiple individual learners can converge to is not altered by any number of agents implementing potential-based reward shaping but their exploration is. This can reduce the training time needed and increase the probability of learning a behaviour of higher performance dependent on the quality of the domain knowledge incorporated.

References

[1] Babes, M., de Cote, E., and Littman, M., Social reward shaping in the prisoner's dilemma, in *Proceedings of the 7th International Joint Conference on Autonomous Agents and Multiagent Systems*, Vol. 3 (2008), pp. 1389–1392.

[2] Balch, T., Learning Roles: Behavioral Diversity in Robot Teams, in *AAAI Workshop on Multiagent Learning* (1997).

[3] Bertsekas, D. P., *Dynamic Programming and Optimal Control (2 Vol Set)* (Athena Scientific, 3rd edition, 2007).

[4] Binmore, K., *Fun and Games - A Text on Game Theory* (D. C. Heath & Co., 1991).

[5] Busoniu, L., Babuska, R., and De Schutter, B., A Comprehensive Survey of Multi-Agent Reinforcement Learning, *IEEE Transactions on Systems Man & Cybernetics Part C Applications and Reviews* **38** (2008) 156.

[6] Claus, C. and Boutilier, C., The dynamics of reinforcement learning in cooperative multiagent systems, in *Proceedings of the National Conference on Artificial Intelligence* (1998), pp. 746–752.

[7] Devlin, S., Grześ, M., and Kudenko, D., Reinforcement learning in robocup keepaway with partial observability, in *IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology, 2009. WI-IAT'09* (2009).

[8] Devlin, S. and Kudenko, D., Theoretical considerations of potential-based reward shaping for multi-agent systems, in *Proceedings of The Tenth Annual International Conference on Autonomous Agents and Multiagent Systems (AAMAS)* (2011).

[9] Eric Wiewiora, G. C. and Elkan, C., Principled methods for advising reinforcement learning agents, in *Proceedings of the Twentieth International Conference on Machine Learning* (2003).

[10] Fudenberg, D. and Tirole, J., *Game Theory* (MIT Press, Cambridge, MA, 1991).

[11] Grześ, M. and Kudenko, D., Multigrid Reinforcement Learning with Reward Shaping, *Artificial Neural Networks-ICANN 2008* (2008) 357–366.

[12] Grześ, M. and Kudenko, D., Plan-based reward shaping for reinforcement learning, in *Proceedings of the 4th IEEE International Conference on Intelligent Systems (IS'08)* (IEEE, 2008), pp. 22–29.

[13] Hu, J. and Wellman, M., Nash Q-learning for general-sum stochastic games, *The Journal of Machine Learning Research* **4** (2003) 1039–1069.

[14] Iscen, A. and Erogul, U., A new perspective to the keepaway soccer: the takers, in *Proceedings of the 7th international joint conference on Autonomous agents and multiagent systems*, Vol. 3 (2008), pp. 1341–1344.

[15] Kalyanakrishnan, S. and Stone, P., Learning complementary multiagent behaviors: a case study, in *RoboCup 2009: Robot Soccer World Cup XIII*, eds. Baltes, J., Lagoudakis, M., Naruse, T., and Ghidary, S., *Lecture Notes in Computer Science*, Vol. 5949 (Springer Berlin / Heidelberg, 2010), pp. 153–165.

[16] Maclin, R. and Shavlik, J., Creating advice-taking reinforcement learners, *Recent Advances in Reinforcement Learning* (1996) 251–281.

[17] Marthi, B., Automatic shaping and decomposition of reward functions, in *Proceedings of the 24th International Conference on Machine learning* (ACM, 2007), p. 608.

[18] Mihaylov, M., Tuyls, K., and Nowé, A., Decentralized Learning in Wireless Sensor Networks, *Adaptive and Learning Agents* (2009) 60–73.

[19] Min, H., Zeng, J., Chen, J., and Zhu, J., A Study of Reinforcement Learning in a New Multiagent Domain, in *IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology, 2008. WI-IAT'08*, Vol. 2 (2008).

[20] Nash, J., Non-cooperative games, *Annals of mathematics* **54** (1951) 286–295.

[21] Ng, A. Y., Harada, D., and Russell, S. J., Policy invariance under reward trans-

formations: Theory and application to reward shaping, in *Proceedings of the 16th International Conference on Machine Learning* (1999), pp. 278–287.

[22] Peters, J., Vijayakumar, S., and Schaal, S., Reinforcement learning for humanoid robotics, in *Proceedings of Humanoids2003, Third IEEE-RAS International Conference on Humanoid Robots* (2003).

[23] Puterman, M. L., *Markov Decision Processes: Discrete Stochastic Dynamic Programming* (John Wiley & Sons, Inc., New York, NY, USA, 1994).

[24] Randløv, J. and Alstrom, P., Learning to drive a bicycle using reinforcement learning and shaping, in *Proceedings of the 15th International Conference on Machine Learning* (1998), pp. 463–471.

[25] Shoham, Y., Powers, R., and Grenager, T., If multi-agent learning is the answer, what is the question?, *Artificial Intelligence* **171** (2007) 365–377.

[26] Stone, P., Kuhlmann, G., Taylor, M. E., and Liu, Y., Keepaway soccer: From machine learning testbed to benchmark, in *RoboCup-2005: Robot Soccer World Cup IX*, eds. Noda, I., Jacoff, A., Bredenfeld, A., and Takahashi, Y., Vol. 4020 (Springer Verlag, Berlin, 2006), pp. 93–105.

[27] Stone, P., Sutton, R. S., and Kuhlmann, G., Reinforcement learning for RoboCup-soccer keepaway, *Adaptive Behavior* **13** (2005) 165–188.

[28] Sutton, R., Generalization in Reinforcement Learning: Successful Examples Using Sparse Coarse Coding, *Advances in Neural Information Processing Systems* (1996) 1038–1044.

[29] Sutton, R. S., *Temporal credit assignment in reinforcement learning*, Ph.D. thesis, Department of Computer Science, University of Massachusetts, Amherst (1984).

[30] Sutton, R. S. and Barto, A. G., *Reinforcement Learning: An Introduction* (MIT Press, 1998).

[31] Tan, M., Multi-Agent Reinforcement Learning: Independent vs. Cooperative Agents, in *Proceedings of the Tenth International Conference on Machine Learning*, Vol. 337 (1993).

[32] Tumer, K. and Khani, N., Learning from actions not taken in multiagent systems, *Advances in Complex Systems (ACS)* **12** (2009) 455–473.

[33] Tumer, K. and Wolpert, D., Collective Intelligence and Braess' Paradox, in *Proceedings of the National Conference on Artificial Intelligence* (2000), pp. 104–109.

[34] Wang, X. and Sandholm, T., Reinforcement learning to play an optimal Nash equilibrium in team Markov games, *Advances in neural information processing systems* (2003) 1603–1610.

[35] Wiewiora, E., Potential-based shaping and Q-value initialization are equivalent, *Journal of Artificial Intelligence Research* **19** (2003) 205–208.

[36] Wolpert, D. and Tumer, K., An introduction to collective intelligence, Technical Report cs.LG/9908014, NASA Ames Research Center (1999).

[37] Wooldridge, M., *An Introduction to MultiAgent Systems* (John Wiley and Sons, 2002).