Embodied Imitation-Enhanced Reinforcement Learning in Multi-Agent Systems

Mehmet D Erbas[1], Alan FT Winfield[2] and Larry Bull[2]

Abstract

Imitation is an example of social learning in which an individual observes and copies another's actions. This paper presents a new method for using imitation as a way of enhancing the learning speed of individual agents that employ a well-known reinforcement learning algorithm, namely Q-learning. Compared to other research that uses imitation with reinforcement learning, our method uses imitation of purely observed behaviours to enhance learning, with no internal state access or sharing of experiences between agents. The paper evaluates our imitation-enhanced reinforcement learning approach in both simulation and with real robots in continuous space. Both simulation and real robot experimental results show that the learning speed of the group is improved.

Keywords

Embodied imitation, reinforcement Q-learning, social learning, multi-agent systems

1 Introduction

Social learning, which enables individuals to learn from others in a community, is an important mechanism for social animals. One of the most important types of social learning is imitation as it allows certain skills and behaviours to be transferred between individuals without language or other complex symbolic communication (Nehaniv and Dautenhahn, 2007). Imitation learning differs from other adaptive learning algorithms that have been used in robotic research, including reinforcement learning (Barto et al., 2004), evolutionary algorithms (Nolfi and Floreano, 2000) and supervised learning (Rumelhart et al., 1986), as learning by imitation is based upon social interactions. Another important aspect of imitation is that the only information transferred between agents is the set of observed actions. An agent imitating another may not know anything about the internal state and structure of the other agent. Therefore imitation is different from other types of learning in that supervision can not be used to directly influence internal processes. This paper presents a simple method for linking reinforcement learning with imitation.

The problem of matching the actuators of the observed robot to the robot's own actuators is presented as the correspondence problem (Nehaniv and Dautenhahn, 2002). A solution to the correspondence problem for robot to robot imitation, for our experimental context, is presented in this paper in the following way: The problem can be divided into two separate issues, (1) how to replicate the observed actions and (2) how to find an appropriate context (or states) for which those actions are meaningful. The first problem is solved by programming intrinsically, that is to say if an agent observes that the other agent turns to its left then goes forward for some time, by using the embodied imitation algorithm developed in this paper, it algorithmically determines that in order to replicate those actions, it has to turn to its left and go forward. The second issue of finding the appropriate state (or context) for which the observed actions are meaningful is solved by the learning process of the imitating agent so that it will infer in what state the observed actions are useful through, for instance, a trial and error mechanism. Thus imitated actions that cause an increase in the performance in a state will be more likely to be learned. Similarly, any negative outcome will make the observed actions unlikely to be associated with that state.

Programming by Demonstration (PbD) (Billard et al., 2008), in which a robot is taught new behaviours by humans or other robots, has been widely studied in Robotics research. With the development of humanoid robots, recent PbD research has an increasingly interdisciplinary approach. For example, it has benefited from examining the neural mechanism for imitation in primates (Billard et al., 2008). As humanoid robots are expected to adapt to changing environments, the flexibility of their control system becomes crucial. As a result, the notion of PbD has been progressively

[1] Istanbul Kemerburgaz University, Faculty of Engineering and Architecture
[2] University of the West of England, Faculty of Environment and Technology

Corresponding Author:
Mehmet D Erbas, Mahmutbey Dilmenler Caddesi, no: 26, Bagcilar, 34217, Istanbul, Turkey
E-mail: mehmet.erbas@kemerburgaz.edu.tr

replaced by a more biologically inspired label: imitation learning. Some example research that used imitation learning in order to train robots are (Gaussier et al., 1998), (Dillmann, 2004), (Breazeal et al., 2005), (Nicolescu and Mataric, 2007), (Calinon and Billard, 2007) and (Guenter & Billard, 2007). There is some research that attempts to use imitation in conjunction with individual learning. Abbel and Ng (Abbeel and Ng, 2004) used an expert in order to learn to perform a task in which the reward function is not known. They tried to obtain the unknown reward function which is supposed to be implicitly followed in the expert's behaviour. A policy is defined as a mapping from states to probability distributions for actions that are possible in a state so that an agent acts according to its policy. The value of a policy is defined as a linear function of the features of the environment and the learning agent has an estimate of the expert's feature expectations. Their algorithm is proven to attain performance close to that of the expert agent. Latzke et al. (Latzke et al., 2006) utilized Q-learning which uses the experience of an expert agent as training data. The imitating agent has full access to the experience of the expert and these experiences are provided as sequences of states and actions along with their rewards. Price and Boutilier (Price and Boutilier, 2003) devised *implicit imitation* to accelerate reinforcement learning. In their method, the observer agent obtains an experience tuple from a mentor agent with each mentor transition. The experience tuple is used to train the observer robot. Bentivegna et al. (Bentivegna et al., 2004) used a modified version of Q-learning in order to improve the performance of an agent through practice. Their algorithm stores the actions of a teacher agent in a number of states. During the practice period, the learner agent selects one of the data entries based on the distance between the state recorded in that entry and the current state. The value of selecting that entry in the current state is then updated, based on the reward observed after the action stored in the selected data entry is performed.

The approach presented in this paper is different from those outlined above for two reasons. Firstly, there is no assumption about the learner agent having access to the internal state, experience or expectations of the imitated agent. The only information transferred to the learner agent are the imitated agent's executed actions. The learner agent does not know anything about in what state these actions were executed or what were the outcomes of these actions. Secondly, the case in which the observed actions are not useful at all is also considered; in this case the learner agent must discriminate between useful and useless actions.

This paper presents and adaptive learning algorithm in which imitation is used as a method for enhancing learning of agents that employ a well-known reinforcement learning algorithm, namely Q-learning. Compared to other research that uses imitation with reinforcement learning, our method uses imitation of purely observed behaviours to enhance learning, with no internal state access or sharing of experiences between agents. As in nature, imitation provides agents with model behaviours that can influence their individual learning. Finding in what state (or context) these model behaviours are useful is determined by the learning process of the imitating agent. As developing the algorithm purely on real robots would take a long time, it is firstly tested in simulation, with simulated agents. Once the algorithm is shown to be effective, it is then tested on real robots. Both simulation and real robot experiment results show that the learning speed of the agents is improved.

2 Imitation-Enhanced Reinforcement Learning

2.1 Simulation Setup

To examine the effects of imitation on learning strategy, an agent that employs Q-learning in order to reach a target location is simulated. For the first set of experiments, the arena in which the agent operates is a 10 by 10 grid world (figure 1). The agent starts from the top-left corner of the arena and, in each time unit, it moves to one of the eight neighbouring cells. One simulation run ends when the agent reaches the goal item, which is placed in the bottom-right corner of the arena.

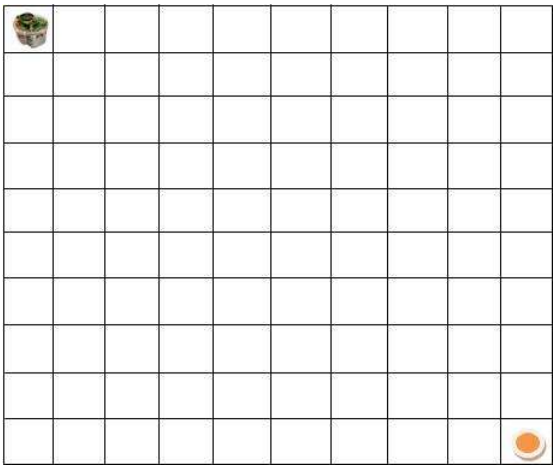

Figure 1. Simulated arena. The arena is a 10 by 10 grid world. The agent is placed in the top-left corner and the goal item is placed in the bottom right corner.

The learning agent uses an $\epsilon - greedy$ algorithm in which at each time step it finds the action that has the highest Q value estimate for its current position. With probability $1 - \epsilon$, it chooses that action and with a probability of ϵ, it chooses a random action. In this way, it updates the Q value for its current state and chosen action:

$$Q(s_t, a_c) = Q(s_t, a_c) + \alpha[r_{t+1} + \gamma argmax_a Q(s_{t+1}, a) - Q(s_t, a_c)] \quad (1)$$

in which a_c is the chosen action, α is the learning ratio, γ is the discount factor and r_{t+1} is the reward for getting to state s_{t+1} (α and γ values are set to 0.2 and 0.7 respectively in the experiments). In one experiment run, the agent gets to the goal state 100,000 times and so the Q values for each state-action pair have adequate time to converge to their final

values. Every 10 time units, the shortest path to the goal item that is learned by the agent is determined by using a greedy action selection method on the current Q values.

2.2 Imitation-Enhanced Reinforcement Learning Algorithm

Only the executed actions are observable and hence copied when an agent imitates another. To simulate this process, for the first experiments a set of actions is given to the learning agent. At each time step, the agent may start to replicate the given set of actions with a probability equal to $p_{imitate}$. When the agent starts to imitate the path, all actions that constitute the path are executed one by one by the agent. If these actions contradict the Q values of the agent, i.e. have a low value, then the agent stops following the imitated path and acts based on its original $\epsilon - greedy$ algorithm. Thus the controller of the agent with imitation is updated as shown in Algorithm 1. But how do we select, $p_{imitate}$, the starting imitation probability? The rest of the section compares 3 different approaches for selecting $p_{imitate}$.

Approach 1 - Fixed value: With the first approach, $p_{imitate}$ is set to a fixed value. In the first experiment run, the agent is given a path that moves it towards the goal item (figure 2) which is a beneficial path, so that the agent has this path in memory and is able to enact it according to its controller. Figure 3 shows the performance of imitation enhanced learning agent with different $p_{imitate}$ values. As can be expected, there is a performance increase in the learning speed of the agent compared to a no-imitation agent. The imitation-enhanced agent finds the shortest path much earlier than the agent with Q-learning only. But what would happen if the path given is not beneficial? To test this scenario, the path in figure 4 is given to the agent in the second experiment run. Figure 5 shows the performance of the agent when this path is given, which is clearly not a beneficial path. A pair-wise t-test[1] reveals that there is a statistically significant performance loss for all β values when the path given is not directing the agent towards the goal item. It is clear that there should be an adaptive $p_{imitate}$ calculation which can determine if an observed path is beneficial or not.

Approach 2 - Path completion test: The agent can also check if an imitated path is typically completed. If the imitated path is being abandoned continuously (because it is rejected by the Q values or because of the physical boundaries of the arena) whenever it is started, it may be a sign of a non-beneficial path. So for the second try, the $p_{imitate}$ value is calculated by:

$n_{completed}$ = *number of times the observed path is completed*

$n_{replicated}$ = *number of times the observed path is replicated*

$$p_{imitate} = \beta \frac{n_{completed} + 1}{n_{replicated} + 1} \quad (2)$$

in which β is a constant that regulates initial imitation probability. Therefore by using this formula, the agent will tend to imitate paths more if they can be completed. Figure 6 shows the performance of the imitation-enhanced learning agent for different β values when the beneficial path in figure 2 is given. Once again there is a clear performance increase in the learning speed of the agent when the imitated path is beneficial. Figure 7 shows the performance of the imitation-enhanced agent when the path in figure 4 is given. Although it is slightly better than the case with a fixed $p_{imitate}$, there is still a statistically significant performance loss for all β values when the path given is not beneficial.

Algorithm 1 Pseudocode for the controller of the agent with one path

Input:
Set $Q(s,a) \leftarrow 0$ for all state-action pairs
A set of observed actions: $actionList \leftarrow \{a_1, a_2, ..., a_n\}$, $actionIndex \leftarrow 0$
Repeat 100.000 *times*
Place the agent in the top-left corner
while $s \neq GoalState$ **do**
  if $actionIndex \neq 0$ **then**
    $actionIndex \leftarrow actionIndex + 1$
    $a \leftarrow actionList[actionIndex]$
    if $ListCompleted(actionList) = true$ **then**
      $actionIndex \leftarrow 0$
    end if
  else
    if $p_{imitate} > random()$ **then**
      $actionIndex \leftarrow 1$
      $a \leftarrow actionList[actionIndex]$
    else
      $a \leftarrow argmax_{a'} Q(s,a')$
      if $\epsilon > random()$ **then**
        $a \leftarrow SelectRandomAction()$
      end if
      $actionIndex \leftarrow 0$
    end if
  end if
  if $actionIndex > 0$ **then**
    if $\exists\ a' : Q(s,a') > Q(s,a)$ **then**
      $a \leftarrow argmax_{a'} Q(s,a')$
      if $\epsilon > random()$ **then**
        $a \leftarrow SelectRandomAction()$
      end if
      $actionIndex \leftarrow 0$
    end if
  end if
  $Q(s,a) \leftarrow Q(s,a) + \alpha[r + argmax_{a'} Q(s',a') - Q(s,a)]$
  $s \leftarrow s'$
end while

[1] A pair-wise t-test checks the hypothesis that, for two data sets S_1 and S_2, the data in $S_1 - S_2$ comes from a normal distribution with mean equal to 0 and unknown variance. The same statistical test is applied to experiment results for comparison.

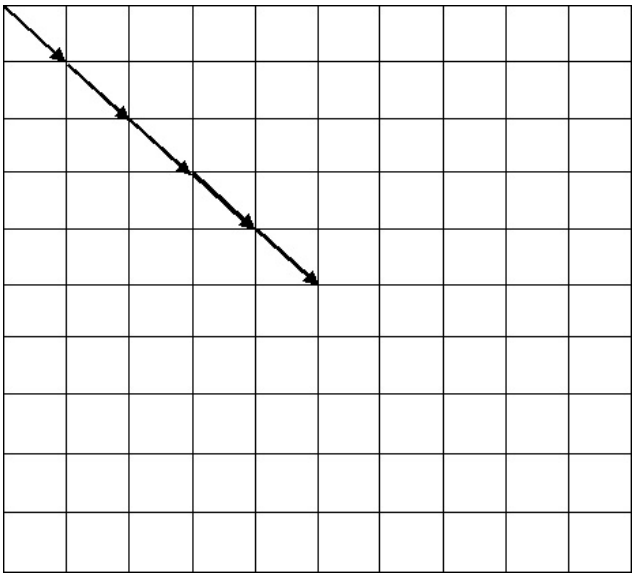

Figure 2. Path given to the agent. It consists of 5 consecutive moves South East. Since by following this path, the agent gets close to the goal item, this path is beneficial for the agent. These moves are shown at relative positions.

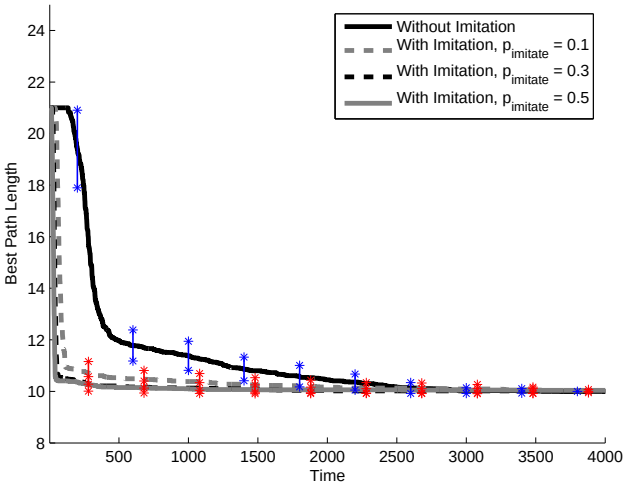

Figure 3. Approach 1: Fixed value, performance of the agent when a beneficial path is given. The results are mean best path length from 100 experiment runs along with 95% confidence intervals (note these are shown only at 10 point intervals). The best path for each agent is calculated every 10 time units by using a greedy algorithm on the current Q values. Time is given in time units.

Approach 3 - Average Q value test: Another approach is to examine the effects of imitation on the Q values of the agent. For this purpose, we check if the observed actions cause any change in Q-value of the related state-action pairs. If they do not, this means that the observed actions are causing the agent to explore some parts of the state-action space which have already converged to their final values or the agent is exploring some parts of the state action-space which do not contribute to its overall performance. If the observed actions do cause a change in Q-values then it is possible that imitation

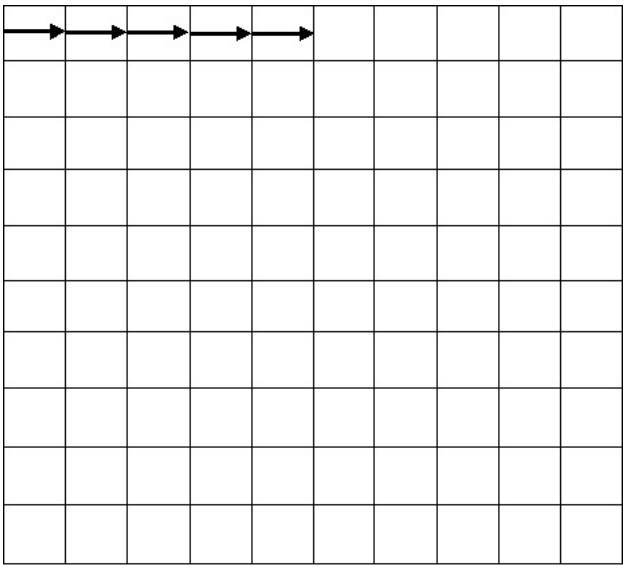

Figure 4. Path given to the agent. It consists of 5 consecutive moves East. Since by following this path, the agent does not get close to the goal item, this path is not beneficial for the agent. These moves are shown at relative positions.

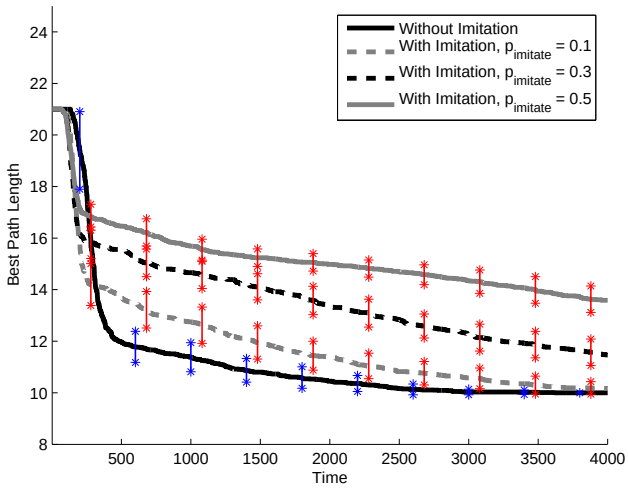

Figure 5. Approach 1: Fixed value, performance of the agent when a non-beneficial path is given. The results are mean best path length from 100 experiment runs along with 95% confidence intervals (note these are shown only at 10 point intervals). The best path for each agent is calculated every 10 time units by using a greedy algorithm on the current Q values. Time is given in time units.

may be beneficial. This is determined as follows: the average Q value is recorded at the start of each imitation sequence and is compared to the average Q value when the imitated path is completed or abandoned. So $p_{imitate}$ is calculated by:

n_{Q+} = *number of times the average Q value increased when the observed path is replicated*

$n_{replicated}$ = *number of times the observed path is replicated*

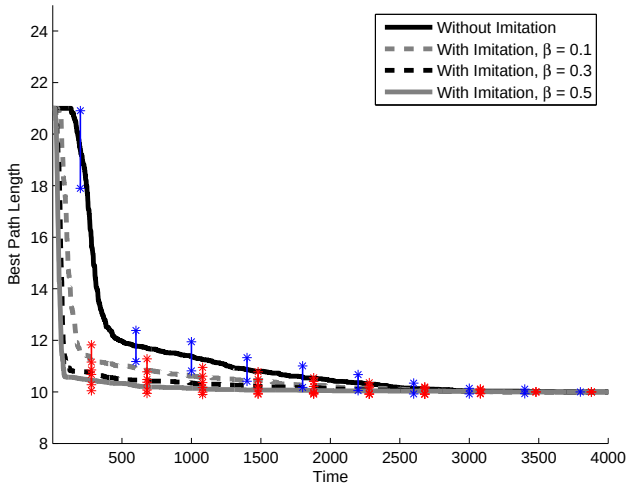

Figure 6. Approach 2: Path completion test, performance of the agent when a beneficial path is given. The results are mean best path length from 100 experiment runs along with 95% confidence intervals (note these are shown only at 10 point intervals). The best path for each agent is calculated every 10 time units by using a greedy algorithm on the current Q values. Time is given in time units.

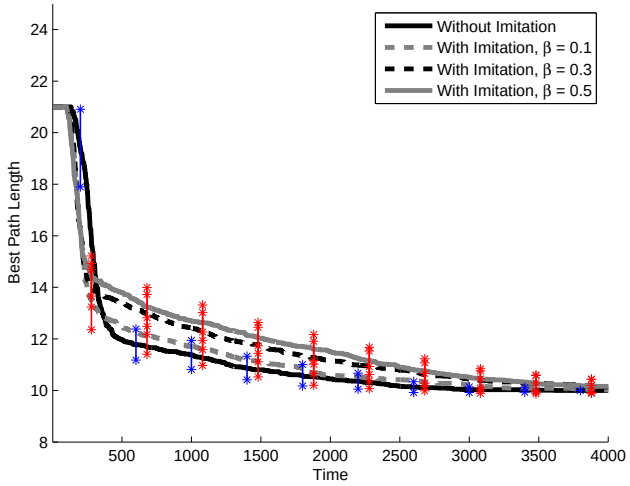

Figure 7. Approach 2: Path completion test, performance of the agent when a non-beneficial path is given. The results are mean best path length from 100 experiment runs along with 95% confidence intervals (note these are shown only at 10 point intervals). The best path for each agent is calculated every 10 time units by using a greedy algorithm on the current Q values. Time is given in time units.

$$p_{imitate} = \beta \frac{n_{Q+} + 1}{n_{replicated} + 1} \tag{3}$$

Figure 8 shows the performance of the imitation-enhanced agent for different β values when the beneficial path in figure 2 is given. Once again there is a clear performance increase in the learning speed of the agent when the imitated path is beneficial. Figure 9 shows the performance of the imitation-enhanced agent when the path in figure 4 is given. As can be seen, the performance loss due to imitating a non-beneficial path is minimal compared to the previous two tries. A pair-wise t-test reveals that when the β value is set to 0.1, the difference between an imitation-enhanced agent and a no-imitation agent is not statistically significant. For higher β values (0.3 and 0.5) there is a statistically significant performance loss but the difference is much smaller than the case with a fixed $p_{imitate}$ and with the path completion test. Therefore, by checking temporal changes in Q values, it is possible to determine if imitating an observed path is beneficial or not.

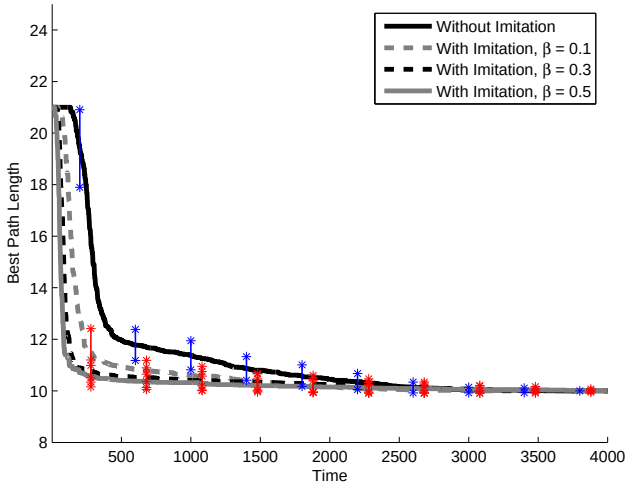

Figure 8. Approach 3: Average Q value test, performance of the agent when a beneficial path is given. The results are mean best path length from 100 experiment runs along with 95% confidence intervals (note these are shown only at 10 point intervals). The best path for each agent is calculated every 10 time units by using a greedy algorithm on the current Q values. Time is given in time units.

As the imitation-enhanced agent is able to determine if a path is beneficial or not by checking temporal changes in Q values, when it observes multiple paths, it should be able to select the ones that are useful and avoid others. For this purpose, the ratio of the times the average Q value increased is calculated for each path i as:

n^i_{Q+} = *number of times the average Q value changed when path i is replicated*

$n^i_{replicated}$ = *number of times path i is replicated*

$$R_i = \frac{n^i_{Q+} + 1}{n^i_{replicated} + 1} \tag{4}$$

R_i is used for both regulating the imitation probability of each path observed and choosing which path to imitate. If the agent has n distinct paths, the probability of choosing path i is calculated by:

$$P^i_{choose} = \frac{R_i}{\sum_{k=1}^{n} R_k} \tag{5}$$

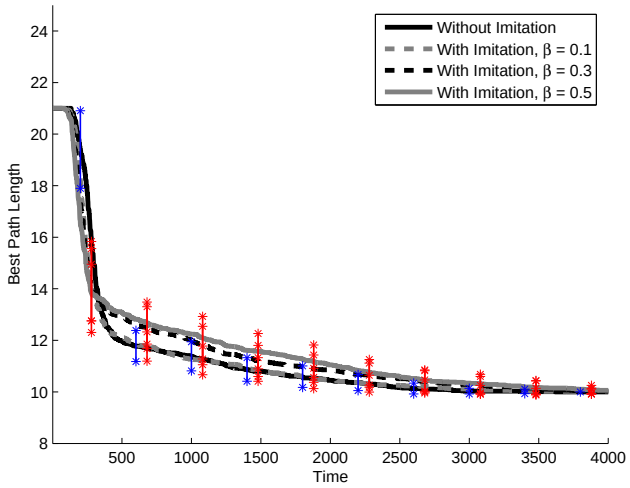

Figure 9. Approach 3: Average Q value test, performance of the agent when a non-beneficial path is given. The results are mean best path length from 100 experiment runs along with 95% confidence intervals (note these are shown only at 10 point intervals). The best path for each agent is calculated every 10 time units by using a greedy algorithm on the current Q values. Time is given in time units.

The probability of imitating path i once it is chosen is:

$$p^i_{imitate} = \beta R_i \tag{6}$$

in which β is a constant that regulates the initial imitation probability. The controller of the agent with multiple paths is updated as shown in Algorithm 2. In appendix A, it is shown that the imitation-enhanced Q-learning algorithm does not violate the conditions for the convergence of Q-learning algorithm.

2.3 Experiments

In order to test the generalization of the algorithm and to make the problem more complex, the next set of experiments are performed in a 100 by 100 grid with an obstacle in the middle (figure 10). Once again the learning agents start from the top-left corner and try to reach the goal item which is placed in the bottom-right corner.

2.3.1 Imitating Predefined Paths

To check if the algorithm is able to select useful paths and improve the learning speed by exploiting them, the learning agent is given 8 paths. These are chosen to each represent 20 consecutive moves in the eight different directions of the compass, as shown in figure 11. The agent has these paths in its memory and it is able to enact them according to its controller. During the experiment, the agent has the following action sets in its memory:

- $Path_1$: E-E-E-E-E-E....-E (20 moves)
- $Path_2$: SE-SE-SE-SE-SE-SE....-SE
- $Path_3$: S-S-S-S-S-S....-S
- $Path_4$: SW-SW-SW-SW-SW-SW....-SW
- $Path_5$: W-W-W-W-W-W....-W
- $Path_6$: NW-NW-NW-NW-NW-NW....-NW
- $Path_7$: N-N-N-N-N-N....-N
- $Path_8$: NE-NE-NE-NE-NE-NE....-NE

If the corresponding R_i values for each path are R_1, R_2,....,R_8, the probability of selecting each path would be equal to $R_i/(R_1+R_2+R_3+R_4+R_5+R_6+R_7+R_8)$. Assume the agent is in state s_t and it selects $Path_1$ and starts it. If there exists $Q(s_t, a) > Q(s_t,' E')$, the path is rejected and the agent moves according to its ϵ-greedy Q-learning. Otherwise the agent starts to follow $Path_1$, making the same check at each step. Whenever a path is rejected or completed, its R_i value is updated.

Figure 12 shows the performance of the imitating agent averaged over 100 experimental runs in comparison with a no-imitation agent. The imitation-enhanced learning agent was able to choose the beneficial paths and improve its performance. A pair-wise t-test reveals that the difference between the two is statistically significant until around time step 15.5m[2]. Figure 13 shows an example shortest path in our arena. Any shortest path consists of a combination of 80 moves south-east, 20 moves east and 20 moves south. Figure 14 shows the R_i values for each of the 8 paths averaged over 100 runs. As can be seen, the path to the SE has the highest R_i value which makes it more likely to be chosen. The paths E and S have relatively higher R_i values compared to the other 5 paths. This confirms that the agent was able to select the 3 paths that are more likely to improve its performance.

2.3.2 Agents Imitating Each Other

a) Agent Copying an Expert Agent: An important aspect of imitation is the possibility of a teacher or expert whose experience can be transferred, by imitation, to other individuals. To simulate an expert in the arena, an agent is programmed to follow the shortest path in Figure 13 indefinitely. Then a second agent is presented in the arena that starts learning and is able to watch and copy the expert agent, following the imitation-enhanced Q-learning algorithm as shown in Algorithm 2. The second agent starts with an empty memory and once every 100 simulation runs, it copies a random consecutive portion of the expert's trajectory. Note that although this is a part of an ideal (i.e. shortest) path, the imitating agent does not know anything about in which state (or location in the arena) these actions are meaningful. The imitating agent can store up to 10 paths. After its memory is full, when a new path is copied, of the paths in its memory the one with the lowest R_i is erased and the new path is saved. Since each experiment consists of 100,000 simulation runs, the agent will imitate and test 1000 portions in each experiment run. Figure 15 shows the performance of the agent imitating

[2]15.5m means 15.500.000 time steps

an expert. As can be seen, the imitation-enhanced learning agent finds the shortest path much earlier than the agent with Q-learning only. A pair-wise t-test reveals that the difference between the two is statistically significant until around time step 15.5m.

Algorithm 2 Pseudocode for the controller of the agent with multiple paths

Input:
Set $Q(s,a) \leftarrow 0$ for all state-action pairs
A set of observed paths $pathList \leftarrow \{Path_1, ..., Path_n\}$
$pathChosen \leftarrow 0$, $actionIndex \leftarrow 0$
$Repeat$ 100.000 $times$
Place the agent in the top-left corner
while $s \neq GoalState$ **do**
  if $pathChosen \neq 0$ **then**
    $actionIndex \leftarrow actionIndex + 1$
    $a \leftarrow pathChosen[actionIndex]$
    if $PathCompleted(pathChosen) = true$ **then**
      $update\ R_{pathChosen}$
      $pathChosen \leftarrow 0$
    end if
  else
    $pathChosen \leftarrow SelectProbabilistically(PathList)$
    if $p_{imitate}^{pathChosen} > random()$ **then**
      $actionIndex \leftarrow 1$
      $a \leftarrow pathChosen[actionIndex]$
    else
      $a \leftarrow argmax_{a'} Q(s,a')$
      if $\epsilon > random()$ **then**
        $a \leftarrow SelectRandomAction()$
      end if
      $pathChosen \leftarrow 0$
    end if
  end if
  if $pathChosen > 0$ **then**
    if $\exists\ a' : Q(s,a') > Q(s,a)$ **then**
      $a \leftarrow argmax_{a'} Q(s,a')$
      if $\epsilon > random()$ **then**
        $a \leftarrow SelectRandomAction()$
      end if
      $update\ R_{pathChosen}$
      $pathChosen \leftarrow 0$
    end if
  end if
  $Q(s,a) \leftarrow Q(s,a) + \alpha[r + argmax_{a'} Q(s',a') - Q(s,a)]$
  $s \leftarrow s'$
end while

b) An Agent Copying an Experienced Agent: In this case, the effect of the existence of a slightly experienced agent instead of an expert is tested. A single agent is trained in the arena for some time (90,000 simulation runs), hence with no-imitation, and a second agent is presented later. The second agent is able to watch and copy the trained agent, following the imitation-enhanced Q-learning algorithm. Note that, even if the experienced agent discovers the best path, it may still make random moves because of its $\epsilon-greedy$ algorithm. Once again the imitating agent copies a new path every 100 simulation runs and has a memory of 10 action lists. Figure 16 shows the results for this experiment. The agent that is watching an experienced (but not expert) agent has a faster learning rate compared to a no-imitation agent. A pair-wise t-test reveals that the difference between the two is statistically significant until around time step 15.5m.

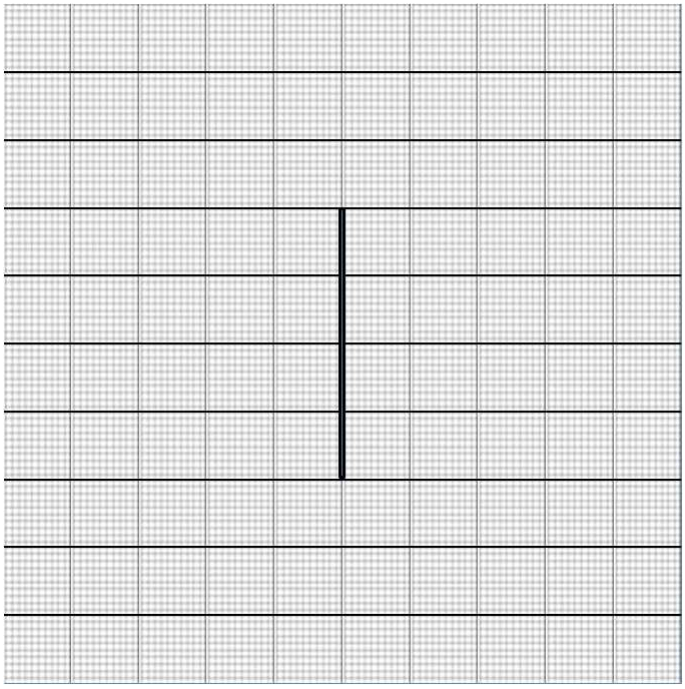

Figure 10. Arena with obstacle: The arena is a 100 by 100 grid world. The agents are placed in the top-left corner and the goal item is placed in the bottom-right corner. There is an obstacle in the middle of the arena.

c) Agents Copying Each Other : But what would happen if there is no expert agent to share its experience? In order to simulate this scenario, two agents are present in the arena, both starting to learn at the same time and able to watch each other. Once again they store 10 action lists and they copy (i.e. imitate) a new path from each other every 100 simulation runs. Figure 17 shows results for this experiment. As can be seen, the imitating agents have a better performance than a no-imitation agent initially. But the difference between them vanishes after 5m time steps. This is due to the fact that the two imitating agents influence each other throughout the experiment. At the start, the one that has slightly better performance is able to improve the other but as time passes both start to converge to the same performance. After some point, their continuous effect on each other causes less improvement which explains why they get similar performance to a no-imitation agent. Pair-wise t-test shows that the imitating agent has a statistically better performance between the time steps 4m and 9.5m. Note also that some runs of the no-imitation

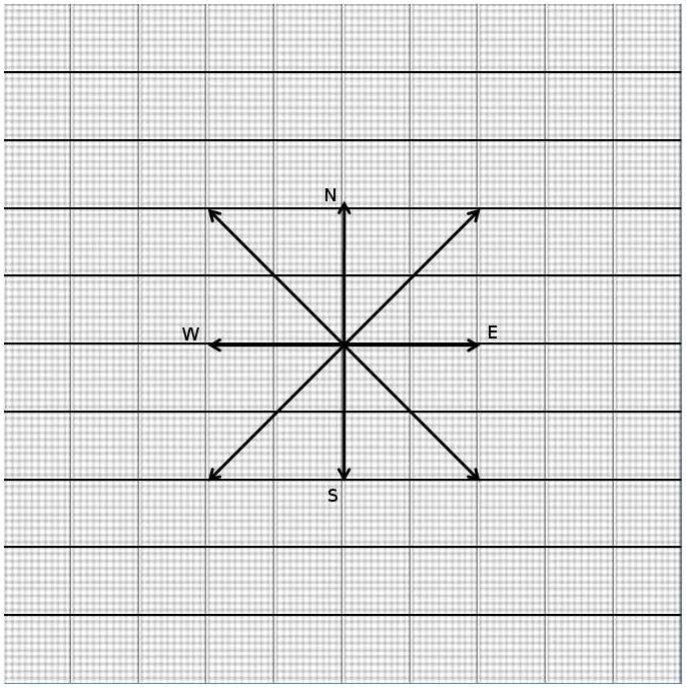

Figure 11. 8 Paths given to the agent. Each list of actions consist of 20 consecutive moves in one direction: E, SE, S, SW, W, NW, N and NE

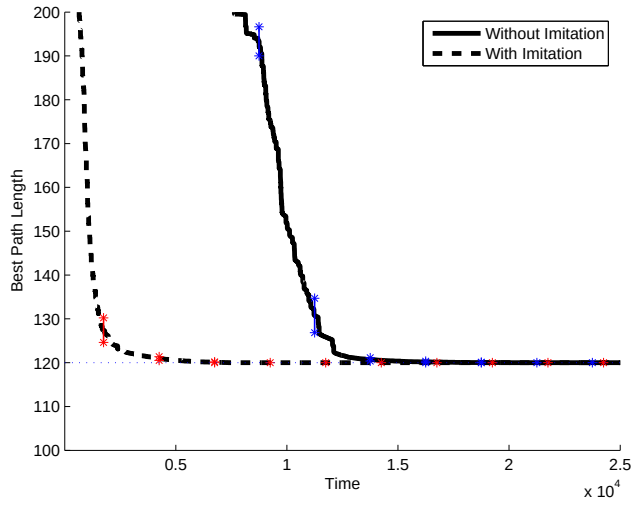

Figure 12. Best path length for imitating (8 paths given) and no-imitation agents. The results are mean best path length from 100 experiment runs along with 95% confidence intervals (note these are shown only at 10 point intervals). The best path for each agent is calculated every 1000 time units by using a greedy algorithm on the current Q values. The β value that regulates the initial imitation probability is set to 0.5 for these experiments. Time is given in 100 time units.

agent converge to a shortest path earlier than the imitating agent which makes the no-imitation agent statistically better between the time steps 12.5m and 19.5m, although the difference between the two is very small.

Figure 18 compares all scenarios: an agent given 8 paths (figure 12), an agent watching an expert (figure 15), two

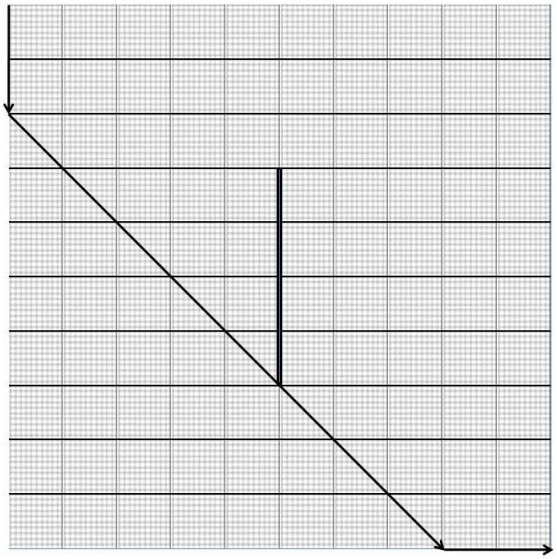

Figure 13. An example shortest path to the goal state. Any shortest path consists of a combination of 80 moves to SE, 20 moves to S and 20 moves to E directions.

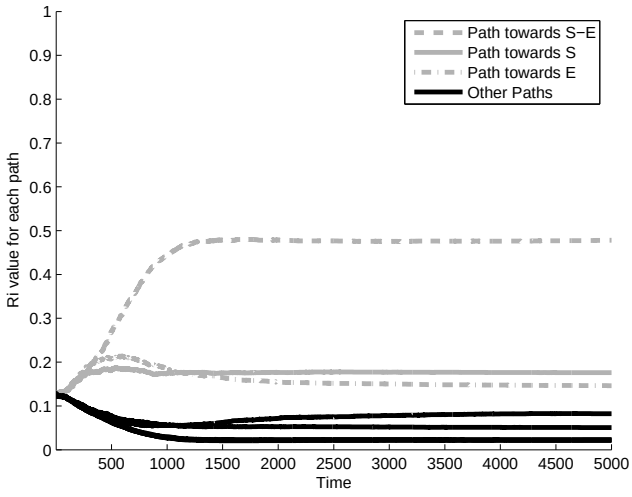

Figure 14. The R_i values for each path given to the agent.

agents watching each other from the beginning (figure 17) and an agent watching an experienced agent (figure 16). As expected the agent copying an expert outperforms all others although the agent that is given 8 paths achieves the same performance as around 5m time steps. The agent that is watching an experienced (but not expert) agent achieves the same performance with these two after around 8m time steps. All three have significantly better performance than the worst case of two agents copying each other.

2.4 Beyond Imitation; Agents Having Access to Each Other's Internal State

Based on the related work discussed in section 1, the perfor-

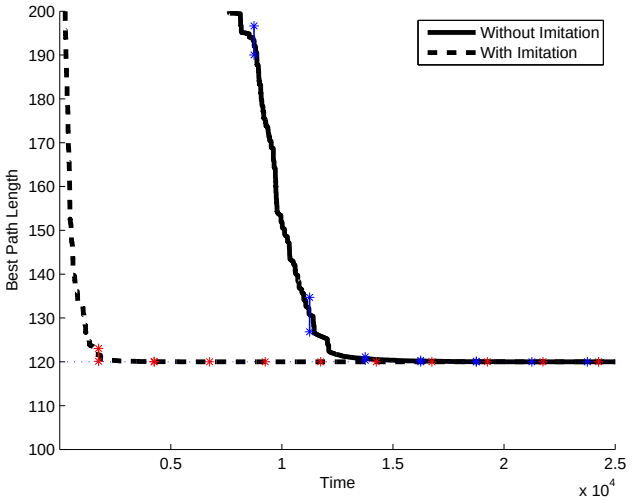

Figure 15. Best path length for imitating (copying an expert) and no-imitation agents. The results are mean best path length from 100 experiment runs along with 95% confidence intervals. The best path for each agent is calculated every 1000 time units by using a greedy algorithm on the current Q values. The β value that regulates the initial imitation probability is set to 0.5 for these experiments. Time is given in 100 time units.

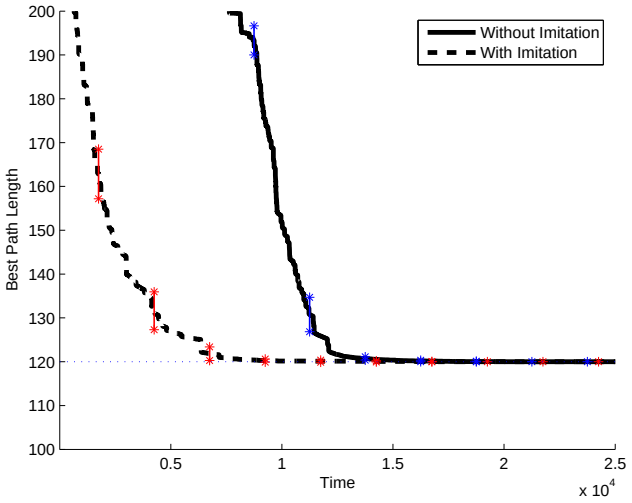

Figure 16. Best path length for imitating (copying an experienced agent) and no-imitation agents. The results are mean best path length from 100 experiment runs along with 95% confidence intervals. The best path for each agent is calculated every 1000 time units by using a greedy algorithm on the current Q values. The β value that regulates the initial imitation probability is set to 0.5 for these experiments. Time is given in 100 time units.

mance of the agents when they are able to access the internal state of the other agent is tested. In this case the agents have complete access to the Q-table of the other agent, i.e. Q values for all state-action pairs. This essentially means that the agent being observed makes available its estimate of the best action for the current state of the observing agent. Therefore the observing agent, instead of copying actions executed by the other agent from potentially any state, determines a path by

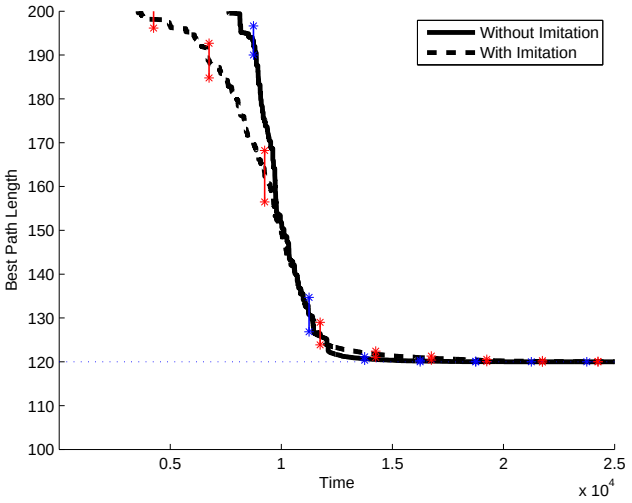

Figure 17. Best path length for imitating (two agents copying each other) and no-imitation agents. The results are mean best path length from 100 experiment runs along with 95% confidence intervals. The best path for each agent is calculated every 1000 time units by using a greedy algorithm on the current Q values. The β value that regulates the initial imitation probability is set to 0.5 for these experiments. Time is given in 100 time units.

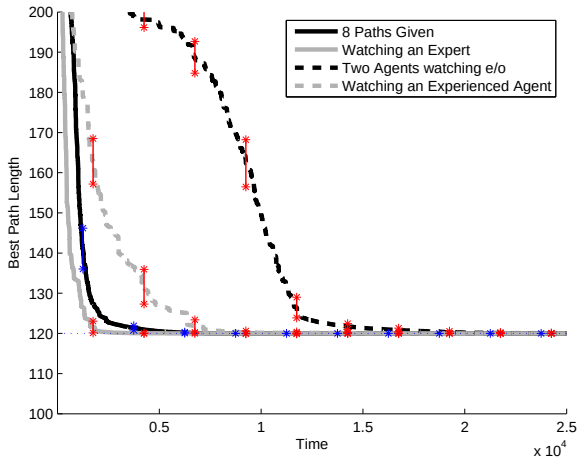

Figure 18. Best path length for 4 scenarios. The results are mean best path length from 100 experiment runs along with 95% confidence intervals. The best path for each agent is calculated every 1000 time units by using a greedy algorithm on the current Q values. Time is given in 100 time units.

repeatedly selecting the action with the highest Q value of the other agent in the current state. Based on the previous mechanism, if the suggested action contradicts its own Q values, i.e. it has a lower value, the observing agent stops imitating.

Figure 19 shows the results of this scheme when copying from an expert. As can be seen, the learning agent becomes optimal after a very short time interval. This is perhaps not unexpected, since the scenario is similar to telepathy. But what if there is no expert? Figure 20 shows the results when two agents, starting to learn at the same time, are able to access

each other's internal state. It almost has the same performance as the $\epsilon - greedy$ Q-learning algorithm as a pair-wise t-test reveals that there is no statistical difference between two.

Figure 21 compares two agents copying each other (figure 17) and two agents copying each other with access to each other's internal state (figure 20). As can be seen, the agent with the imitation-enhanced Q-learning algorithm learns more quickly and a pair-wise t-test reveals that the agent utilising imitation is statistically better than the other scenario between time steps 4m to 9.5m. The reason for the difference can be explained as follows: The agent with the state access receives state specific information from the other agent. For this information to be useful, it is required that the observed agent has visited that specific state and have some Q-values assigned to the state-action pairs in that state. If not, the observed agent essentially suggests a random action to the imitating agent. But with observed imitation, the only information that is transferred to the imitating agent is the set of performed actions. Further, the imitating agent is able to test the utility of these sets of actions in different states and determine if they are useful. Thus, it can be deduced that there is more scope for exploration using the observed action sequences.

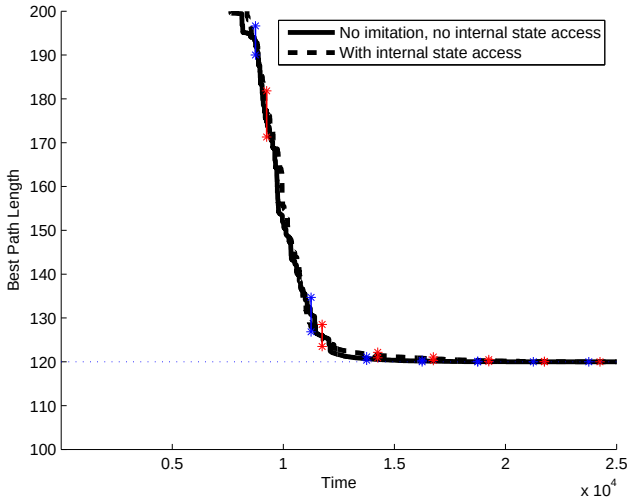

Figure 20. Best path length for an internal-state-access (two agents using each other's Q values) agent and no-imitation/no-internal-state-access agents. The results are mean best path length from 100 experiment runs along with 95% confidence intervals. The best path for each agent is calculated every 1000 time units by using a greedy algorithm on the current Q values. Time is given in 100 time units.

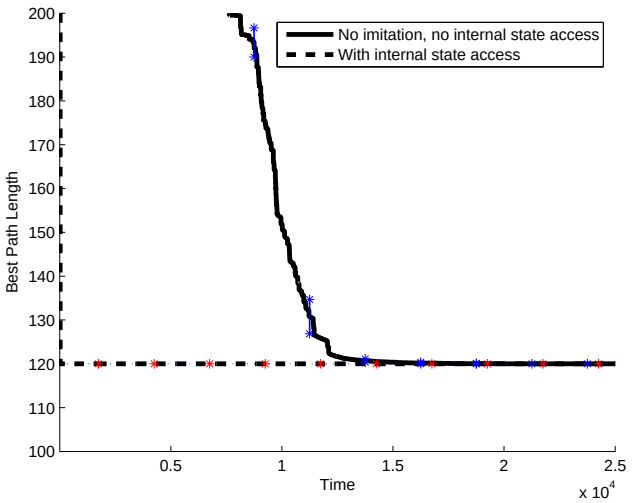

Figure 19. Best path length for an internal-state-access (accessing an expert agent's Q values) agent and no-imitation/no-internal-state-access agents. The results are mean best path length from 100 experiment runs along with 95% confidence intervals. The best path for each agent is calculated every 1000 time units by using a greedy algorithm on the current Q values. Time is given in 100 time units.

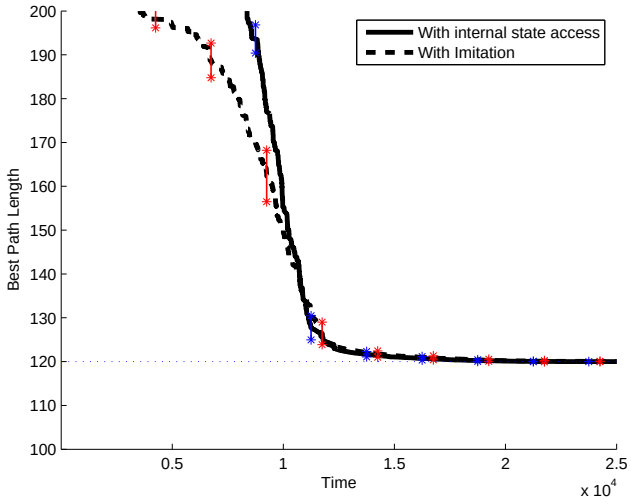

Figure 21. Best path length for an agent accessing another agent's Q values and imitating (two agents copying each other) agent. The results are mean best path length from 100 experiment runs along with 95% confidence intervals. The best path for each agent is calculated every 1000 time units by using a greedy algorithm on the current Q values. Time is given in 100 time units.

3 Imitation-Enhanced Reinforcement Learning in Continuous Space

The most important simplification in the previous section is that the algorithm was tested in simulation. The agents do not possess any physical structure and they interact based on an abstract model of perfect imitation by copying each other's trajectory. In this section, the imitation-enhanced Q-learning algorithm is tested on real robots. The robots use an extended version of Q-learning that is adapted for continuous time. The effects of embodied imitation on the learning speed of robots is evaluated in experiments as the robots learn a simple task.

There are a number of mechanisms proposed in order to use reinforcement learning methods in continuous time and space (Santamaria et al., 1998), (Smart and Kaelbling, 2000), (Doya, 2000), (Strosslin and Gerstner, 2006), (Peters and Schaal, 2006). Although we have a continuous-time problem, the learning robot has to make choices only infrequently. Such

systems are called *semi-Markov Decision Processes* (SMDP) (Sutton and Barto, 1998). They can be treated as a Markov process by taking the reward on each discrete transition as an integral of the cumulative reward on the continuous time interval passed before that transition. Bradtke and Duff (Bradtke and Duff, 1995) introduced algorithms, based on reinforcement learning, adopted to the solution of SMDPs. They applied the SMDP version of Q-learning to the problem of determining the optimal control for a queuing system. Using a similar approach, Crites and Barto (Crites and Barto, 1996) applied the SMDP version of Q-learning, with some additional constraints that provide prior knowledge to the system, to an elevator dispatching task. They used a neural network for function approximation to represent the action-value function. Since we wanted to apply the imitation-enhanced Q-learning algorithm developed in previous section to continuous time and space, and our system has the property of SMDP, we used the method that is developed by Crites and Barto, by dividing our arena into 10x10 grids. Assume that the robot is in state s and it selects action a at time t_1. If the next state transition will happen at time t_2 and the next state will be s', then the update for the Q value for the semi-Markov decision process is calculated by (Crites and Barto, 1996):

$$Q(s,a) = Q(s,a) + \alpha[\int_{t1}^{t2} e^{-\beta(\tau - t_1)} r_\tau d\tau + e^{-\beta(t_2 - t_1)} argmax_{a'} Q(s',a') - Q(s,a)] \quad (7)$$

in which $e^{-\beta(t_2-t_1)}$ acts as a variable discount factor, similar to the γ parameter in the discrete time formula, and it depends on the time between state transitions. Each time the robot makes a decision and moves to its next state, the Q value of its state-action pair is updated based on this formula.

3.1 Hardware Setup

The robots that are used in the experiments are *e-puck* miniature robots (Mondada et al., 2009), 7 cm in diameter and 5 cm in height. They are equipped with 2 stepper motors, two wheels of 41 mm diameter, 8 proximity sensors, a CMOS image sensor, an accelerometer, a microphone, a speaker and a ring of coloured LEDs. Their on-board battery provides 3 hours of autonomy. The original e-puck robot lacks the computational power needed for the image processing required for imitation. To overcome this limitation, the robots are enhanced with a Linux extension board (Liu and Winfield, 2011) based on the 32-bit ARM9 micro-controller with the Debian/Linux system installed. The board has a USB extension port, used to connect a wireless network card, and is equipped with a MicroSD card slot. These additions to the standard e-puck robot offer some benefits, including increased processing power and increased memory. The robots are also fitted with coloured 'skirts' to enable them to see each other using their built-in image sensors. The experiments are performed in an arena measuring 3 m x 3 m. A vision-tracking system provides high-precision position tracking and a dedicated swarm server combines the data from the tracking system and the internal data from robots for later analysis. Each robot is also fitted with a tracking 'hat' which provides a matrix of pins holding unique patterns of reflective markers that allow the tracking system to uniquely identify and track each robot (Figure 22).

Figure 22. An e-puck with Linux board fitted in between the e-puck motherboard (lower) and the e-puck speaker board(upper). Note both the red skirt, and the yellow hat that provides a matrix of pins for the reflective spheres which allow the tracking system to identify and track each robot.

3.2 Movement Imitation Algorithm

In this section, an embodied robot-to-robot movement imitation algorithm is implemented on the Linux extended e-puck robots. Each robot is able to track and copy the other robot's movement patterns. The algorithm completely depends on the visual data coming from the image sensor of the robots; no other type of communication is allowed between the robots.

There are 3 main stages in the imitation algorithm:

- Frame processing: While observing captured visual frames, the observing robot tracks the movement of the demonstrator robot. As stated above, the robots are fitted with coloured skirts; by determining the size and location of the skirt on the demonstrator robot, the observing robot estimates the relative position of the demonstrator and stores this information in a linked list of positions. In this way, up to 5 frames per second are processed.
- Data processing: After the demonstrator's movement pattern is completed, the observer robot processes the linked list of positions using a regression line-fitting method to convert the estimated positions into straight line segments.
- Pattern replication: The straight line segments and their intersections are converted into a sequence of motor

commands (moves and turns).

In this way, the observing robot replicates the pattern demonstrated by the demonstrator robot. Further information on the embodied imitation algorithm can be found at (Winfield and Erbas, 2011).

3.3 Experimental Setup

To examine the effects of embodied imitation on learning speed, robots that employ imitation-enhanced Q-learning to reach a target location are used. The robot arena is divided into multiple compartments, of size 120 cm x 120 cm and a robot resides in each of them (figure 23). Figure 24 shows two robots in their separate compartments performing individual learning. Similar to the previous section's settings, each compartment is divided into 10x10 grids and a look-up table method is used to store Q values for each state-action pair. The tracking system acts as a GPS server and so it broadcasts the location of each robot over the network. Upon receiving this information, the robots determine their current state and decide on their next action. So the robots use the position information to determine where they themselves are (not to determine the movement pattern of an observed robot). They start from the top-left corner of their compartment and they can move to eight different directions of the compass to get to their next state. An experiment track ends when the robot reaches its target location which is placed in the bottom-right corner of the compartment. One experiment run ends when 15000 moves are enacted by the robot. Compared to a simulation with the same settings, the real robot experiments need a much longer time period as an experimental run requires around 10 hours.

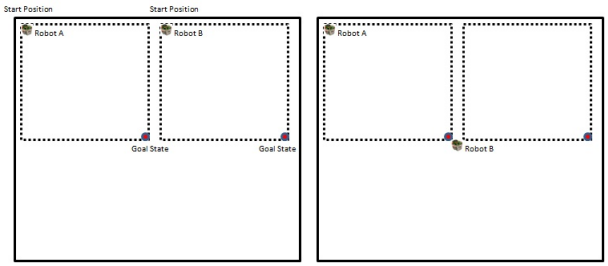

Figure 23. The robot arena is divided into multiple compartments and a robot resides in each of them. The figure on the left shows two robots, robot A and robot B, in their own compartments, performing their individual learning. In the figure on the right, robot B is watching and copying actions performed by robot A.

The learning robot uses an ϵ-greedy algorithm in which, at every state transition, it finds the action that has the highest Q value estimate for its current state. With probability of $1 - \epsilon$, it chooses the action with the highest Q value and, with probability of ϵ, it chooses a random action. In this way, it updates the Q value for its current state and selected action using the update formula explained in the previous section. After every 10 moves, the shortest path that is learned by the robot is determined by using a greedy action selection method on the current Q values.

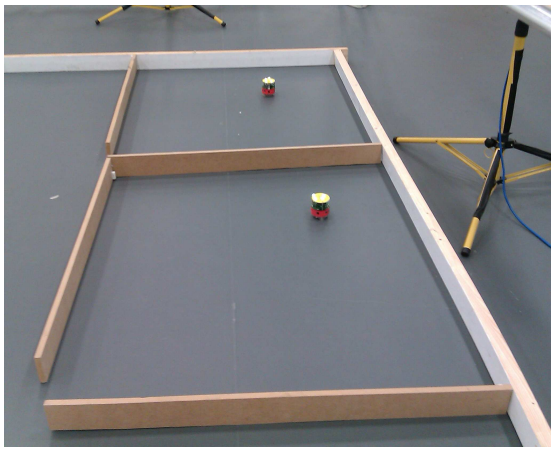

Figure 24. Two robots performing individual learning in their separate compartments.

The robots are able to watch and copy the actions that are enacted by other robots, using the embodied imitation algorithm presented in the previous section. In accordance with the experimental scenario in the previous section, each time a robot needs to observe another robot, by human intervention, it is physically moved to the bottom-right corner of the compartment of the robot that it plans to imitate. The imitating robot then watches the actions enacted by the observed robot for approximately 30 seconds. Once its observation finishes, the observing robot is returned to its compartment and continues its individual learning.

3.4 Experiments

3.4.1 Learner Robot Copying an Expert Robot

In the first set of experiments, the learning speed of a robot, that is able to watch and copy the actions that are executed by an expert robot, using the embodied imitation algorithm presented in section 3.2, is examined. In order to do that, a robot (expert) is programmed to follow the shortest path to the target location indefinitely in its compartment. A second robot (learner) is then placed in another compartment and starts learning using the imitation-enhanced Q-learning method. The pseudo-code of the controller of the learning robot is given in algorithm 3. At the beginning of the experiment, the learning robot moves to the bottom-right corner of the compartment of the expert robot and randomly copies 5 consecutive moves of the expert. It then moves back to its own compartment and starts its individual learning. At the end of every 10 experiment tracks, the learning robot copies a new path of the expert robot[3]. The learning robot has a memory of 10 paths and once filled, no more paths are copied.

Figure 25 shows the performance of robots imitating an expert, in comparison with robots without imitation. As can

[3]The number of consecutive moves to be copied and how often a new path will be copied is chosen to be proportional to the simulation settings in section 2.3.

Algorithm 3 Pseudocode for the controller of the robot

Input:
Set $Q(s,a) \leftarrow 0$ for all state-action pairs
$pathList \leftarrow \emptyset, pathChosen \leftarrow 0$, $actionIndex \leftarrow 0, trackCounter \leftarrow 0$
$moveCounter \leftarrow 0, s \leftarrow \emptyset$
while $moveCounter < 15000$ **do**
  if $trackCounter\%10 = 0$ && $trackCounter < 100$ **then**
    $newPath \leftarrow CopyNewPath()$;
    $PathList \leftarrow AddNewPath(newPath)$;
  end if
  $trackCounter \leftarrow trackCounter + 1$
  $MoveToTopLeftCorner()$
  $s \leftarrow GetState()$
  while $s \neq GoalState$ **do**
    if $pathChosen \neq 0$ **then**
      $actionIndex \leftarrow actionIndex + 1$
      $a \leftarrow pathChosen[actionIndex]$
      if $PathCompleted(pathChosen) = true$ **then**
        $update\ R_{pathChosen}$
        $pathChosen \leftarrow 0$
      end if
    else
      $pathChosen \leftarrow SelectProbabilistically(PathList)$
      if $p_{imitate}^{pathChosen} > random()$ **then**
        $actionIndex \leftarrow 1$
        $a \leftarrow pathChosen[actionIndex]$
      else
        $a \leftarrow argmax_{a'} Q(s,a')$
        if $\epsilon > random()$ **then**
          $a \leftarrow SelectRandomAction()$
        end if
        $pathChosen \leftarrow 0$
      end if
    end if
    if $pathChosen > 0$ **then**
      if $\exists\ a' : Q(s,a') > Q(s,a)$ **then**
        $a \leftarrow argmax_{a'} Q(s,a')$
        if $\epsilon > random()$ **then**
          $a \leftarrow SelectRandomAction()$
        end if
        $update\ R_{pathChosen}$
        $pathChosen \leftarrow 0$
      end if
    end if
    $Q(s,a) = Q(s,a) + \alpha[\int_{t1}^{t2} e^{-\beta(\tau - t_1)} r_\tau d\tau + e^{-\beta(t_2 - t_1)} argmax_{a'} Q(s',a') - Q(s,a)]$
    $PerformAction(a)$
    $s \leftarrow GetState()$
    $moveCounter \leftarrow moveCounter + 1$
  end while
end while

be seen, the imitation-enhanced robot finds the shortest path much earlier than the robot with Q-learning only. During experiments, copying errors occasionally occur but the learning robot is able to exploit the information coming from the expert as the robots are able to discriminate between useful and useless observed behaviours. The effect of copying errors is further discussed in section 3.5.

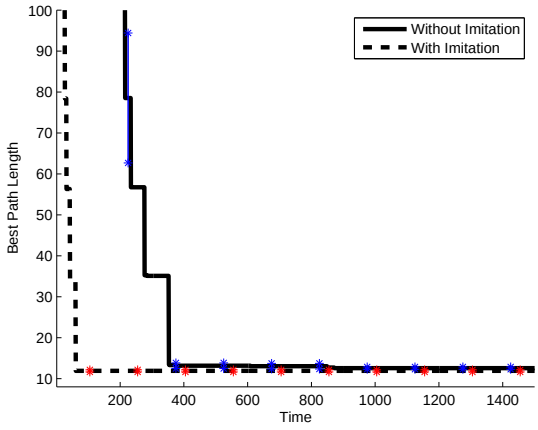

Figure 25. Best path length for Q-learning with imitation (copying an expert) and Q-learning with no-imitation robots. The results are mean best path length from 4 experiment runs along with 95% confidence intervals. The best path for each robot is calculated every 10 state transitions by using a greedy action selection algorithm on the current Q values. The β value that regulates the initial imitation probability is set to 0.5 for these experiments. The distance between two subsequent data points in the plot corresponds to the time needed for 10 state transitions of the robots.

3.4.2 Two Robots Copying Each Other

In the second set of experiments, two robots are placed in separate compartments, both starting to learn at the same time and able to watch each other. At the beginning of the experiment, robots, in turn, are moved to the bottom-right corner of the compartment of the other robot and copy 5 consecutive moves of the other robot. Then, they are moved back to their own compartment and start their individual learning. Once again, at the end of every 10 experiment tracks, the robots copy a new path of the other robot. They have a memory of size 10 paths and once it is filled, no more paths are copied. Figure 26 shows results for this experiment. In accordance with the results from simulation with the same settings (figure 17), the imitating robots have a higher learning speed, on average, as they are able to achieve better solutions earlier than the robots with Q-learning only. However, as expected, the difference between the two cases is smaller compared to the previous set of experiments. A close examination of the results reveals that if one of the robots makes a better start in its individual learning, by chance, then it is able to achieve better solutions sooner than the other robot. As that robot improves its performance, the other robot is able to learn from it by imitation. Therefore, imitation has a positive effect on the collective success of the robots in this scenario. On the other

hand, if none of the robots make a good start, imitation does not improve the learning speed of robots as there is nothing to be gained by copying the other robot. The chance effect explains the larger variations in the experimental results.

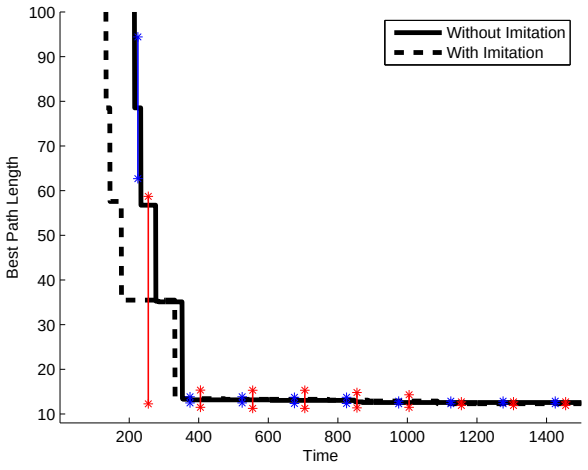

Figure 26. Best path length for Q-learning with imitation (two robots imitating each other) and Q-learning with no-imitation robots. The results are mean best length from 4 experiment runs along with 95% confidence intervals. The best path for each robot is calculated every 10 state transitions by using a greedy action selection algorithm in the current Q values. The β value that regulates the initial imitation probability is set to 0.5 for these experiments. The distance between two subsequent data points in the plot corresponds to the time needed for 10 state transitions of the robots.

During these experiments, the robots are able to choose the paths that are more beneficial to their individual learning, by using the selection method that checks temporal changes in the Q values. Figure 27 shows two example paths that are copied and then enacted the highest number of times during an experimental run. As imitating these paths improves the performance of the robot, they are more likely to be selected and they have the highest R_i values. Figure 28 shows two example paths that are copied and then enacted the smallest number of times for the same experimental run. These two paths have the lowest R_i values which make them less likely to be selected.

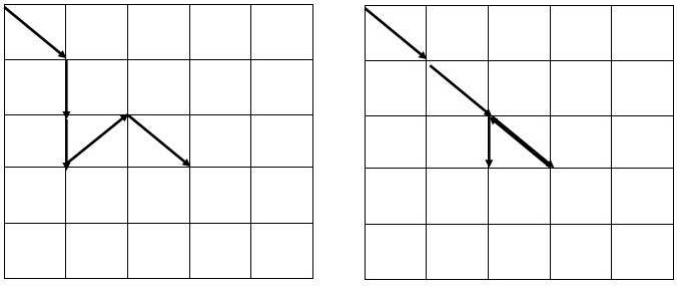

Figure 27. Two paths that are copied and then enacted highest number of times during an experimental run.

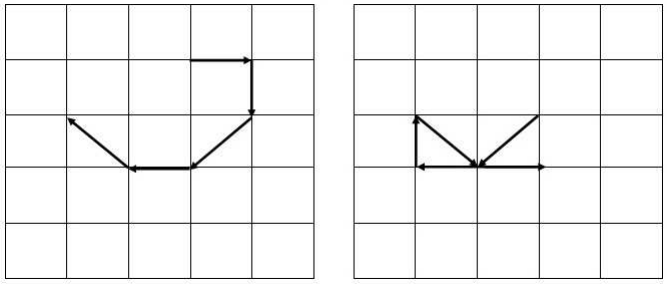

Figure 28. Two paths that are copied and then enacted lowest number of times during an experimental run.

3.5 On the Effects of Embodied Imitation

As stated above, the robots use embodied imitation in which they utilise their on-board sensors to watch and copy each other's actions. Copying errors that arise from embodiment provide variation in the imitation which cause copied paths to differ from their originals. Figure 29 shows two example paths that were copied and then enacted the largest number of times during an experimental run, together with the original paths. As can be seen, they are not perfect copies. As the most enacted movement patterns were the result of imperfect imitations (in some experimental runs), it can be deduced that these errors may have a positive effect on the learning of robots. Variations that arise from embodiment may have a functionality somewhat similar to the that of mutations in a genetic algorithm in that they allow different movement patterns to emerge. If these variations are by chance beneficial, they can be selected by the path selection method of the imitation-enhanced Q-learning algorithm, as in fact observed in some experimental runs. The effects of embodied imitation on the emergence of shared behaviours in a group of robots is further explored in (Erbas and Winfield, 2011).

4 Conclusion and Discussion

This paper has presented a method for using imitation as an enhancing factor to increase the speed of learning for the well known reinforcement learning algorithm, Q-learning. It was shown that information gained by imitation could be used within the Q-learning algorithm. By checking the temporal changes in Q values, it is possible to determine if imitating some observed paths is beneficial or not. The algorithm was first tested in a simple abstract model in simulation and it was shown that the imitation-enhanced Q-learning agent learned faster than the same Q-learner without imitation. It is important to note that, compared to other research that uses imitation with reinforcement learning, our method uses imitation of purely observed behaviours to enhance learning with no internal state access or sharing of experiences between agents. The only information transferred between agents is the imitated agent's executed actions.

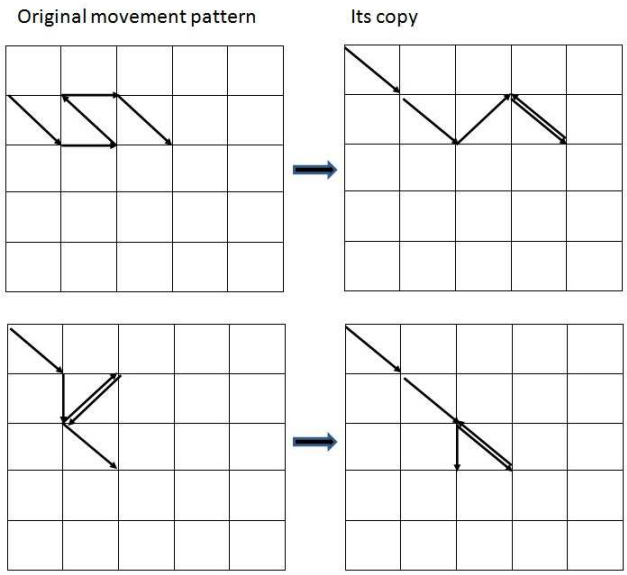

Figure 29. Two example movement patterns that were copied and then enacted the largest number of times. The shape on the left shows the original movement pattern of the observed robot and the figure on the right shows its perceived copy by the imitating robot.

Finding in what state or context these actions are meaningful is determined by the learning process of the imitating agent.

The algorithm was tested on real robots and it was shown that information gained by imitation could be used within the Q-learning algorithm in a continuous time learning task. Robots were able to exploit imitated actions of an expert or experienced robot to improve their individual learning speed. In the case of two robots copying each other, imitation improved the collective learning speed of the robots given that, generally, one of the robots made a better start than the other. The effects of embodied imitation on learning speed was also examined. It was shown that variations that result from copying errors may allow novel solutions to emerge. If these variations lead to a beneficial movement pattern, it could be selected by the path selection method of the imitation-enhanced learning algorithm. The variations that result from embodied imitation can only be examined if real robots are used instead of simulated agents. Therefore, there is a clear value in examining embodied imitation in real robots rather than relying entirely on simulation.

As it was explained in section 3, applying reinforcement learning methods to tasks in continuous time and space is an open research area and there are many mechanisms for different applications. Since the task our robots solved could be formalised as an SMDP, we used the method that was developed by Crites and Barto by dividing the compartment of each robot into 10x10 grids. This simple solution allowed us to test the imitation-enhanced Q-learning algorithm on real robots. The algorithm can be further extended so that it could be used for tasks in continuous time without dividing the robot arena into grids and so allowing any type of move and turn in continuous time and space.

Acknowledgements

This work was supported by EPSRC research grant: EP/E062083/1. The authors are grateful for the very helpful comments of the anonymous reviewers.

APPENDIX

A. Convergence Proof of Imitation-Enhanced Reinforcement Learning

Assume that we have a Markov decision process that is denoted by the tuple (S,A,P,R) in which S is the finite set of possible states, A is the finite set of possible actions, P is the transition probabilities, and R is the reward function. Q-learning converges to optimal Q function **?** if

$$\sum_t \alpha_t(s,a) = \infty \;\; and \;\; \sum_t \alpha_t^2(s,a) < \infty \tag{8}$$

for all $s \in S$ and for all $a \in A$, in which α is the learning rate. Since $0 \leq \alpha_t(s,a) < 1$, all state-action pairs should be visited infinitely often. We will prove that this prerequisite is not validated in imitation-enhanced Q-learning algorithm. Assume that the observed path i constitutes policy π_i and $\pi_i(0) = a_i^0$ (the first action given by the policy π_i). For each $s \in S$,

if $argmax_a Q(s,a) = 0$, then

$$p(a_i^0 : s) = p_i^{start} + \frac{1 - p_i^{start}}{n(A)} \tag{9}$$

which gives that if $p_i^{start} < 1$, all state-action pairs in this state will be visited.

if $argmax_a Q(s,a) > 0$,then

if $\exists a'$ such that $Q(s,a') > Q(s,a_i^0)$, then the a_i^0 is rejected and the agent acts in its ϵ-greedy algorithm.

if $argmax_a Q(s,a) = Q(s,a_i^0)$, then

$$p(a_i^0 : s) = p_i^{start} + (1 - p_i^{start}) * (1 - \epsilon) + (1 - p_i^{start}) * \frac{\epsilon}{n(A)} \tag{10}$$

which gives that if $p_i^{start} < 1$, all state-action pairs will be visited. Therefore, the actions that are imitated are reinforced by the algorithm but it does not cause any of the state-action pairs not to be visited if the initial imitation starting probability, $p_{imitate}$, is smaller than 1.

REFERENCES

Abbeel, P. and Ng, A. Y. (2004). Apprenticeship learning via inverse reinforcement learning. In *Proceedings of ICML 2004*.

Barto, A. G., Bradtke, S. J., and Singh, S. P. (2004). Learning to act using real-time dynacim programming. *Artificial Intelligence*, 6:105–122.

Bentivegna, D. C., Atkeson, C. G., and Cheng, G. (2004). Learning from observation and practice using behavioral primitives: marble maze. *Robotics*, pages 551–560.

Billard, A., Calinon, S., Dillmann, R., and Schaal, S. (2008). Robot programming by demonstration. *Springer Handbook of Robotics*, pages 1371–1394.

Bradtke, S. J. and Duff, M. O. (1995). Reinforcement learning methods for continuous-time markov decision problems. In Tesauro, G., Touretzky, D., and Leen, T., editors, *Advances in Neural Information Processing Systems: Proceedings of the 1994 Conference*, pages 393–400. MIT Press, Cambridge, MA.

Breazeal, C., Buchsbaum, D., Gray, J., Gatenby, D., and Blumberg, B. (2005). Learning from and about others: Towards using imitation to bootstrap the social understanding of others by robots. *Artificial Life*, 11(1-2):31–62.

Calinon, S. and Billard, A. (2007). Incremental learning of gestures by imitation in a humanoid robot. In *Proceedings of ACM/IEEE International Conference on Human-Robot Interaction*, pages 255–262.

Crites, R. H. and Barto, A. G. (1996). Improving elevator performance using reinforcement learning. In Touretzky, D., Mozer, M., and Hasselmo, M., editors, *Proceedings of the Neural Information Processing Systems 8*.

Dillmann, R. (2004). Teaching and learning of robot tasks via observation of human performance. *Journal of Robotics and Autonomous Systems*, 47(2-3):109–116.

Doya, K. (2000). Reinforcement learning in continuous time and space. *Neural Computation*, 12:219–245.

Erbas, M. D. and Winfield, A. F. T. (2011). On the emergence of structure in behaviours evolved through embodied imitation in group of robots. In *Proceedings of the Eleventh European Conference on the Synthesis and Simulation of Living Systems (ECAL 2011)*, pages 219–226.

Gaussier, P., Moga, S., Banquet, J. P., and Quoy, M. (1998). From perception-action loop to imitation process: a bottom-up approach of learning by imitation. *Applied Artificial Intelligence*, 7(1):701–729.

Latzke, T., Behnke, S., and Benewitz, M. (2006). Imitative reinforcement learning for soccer playin robots. In *Proceedings of 10th International Robocup Symposium*.

Liu, W. and Winfield, A. F. T. (2011). Open-hardware e-puck linux extension board for experimental swarm robotics research. *Microprocessors and Microsystems*, 35(1).

Mondada, F., Bonani, M., Raemy, X., Pugh, J., Cianci, C., Klaptocz, A., Magnenat, S., Zufferey, J. C., Floreano, D., and Martinoli, A. (2009). The e-puck, a robot designed for education in engineering. In *9th Conference on Autonomous Robot Systems and Competitions*, page 59:65.

Nehaniv, C. L. and Dautenhahn, K. (2002). The correspondence problem. In Nehaniv, C. L. and Dautenhahn, K., editors, *Imitation in Animals amd Artifacts*, pages 41–61. MIT Press.

Nehaniv, C. L. and Dautenhahn, K. (2007). *Imitation and Social Learning in Robots, Humans and Animals*. Cambridge University Press.

Nicolescu, M. and Mataric, M. J. (2007). Task learning through imitation and human-robot interaction. In Nehaniv, C. L. and Dautenhahn, K., editors, *Imitation and Social Learning in Robots, Humans and Animals*, pages 407–424. Cambridge University Press.

Nolfi, S. and Floreano, D. (2000). *Evolutionary Robotics*. MIT Press.

Peters, J. and Schaal, S. (2006). Policy gradient methods for robotics. In *Proceedings of the International Conference on Artificial Neural Networks*.

Price, B. and Boutilier, C. (2003). Accelerating reinforcement learning through implicit imitation. *Artificial Intelligence Research*, 19:569–629.

Rumelhart, D. E., Hinton, G. E., and William, R. J. (1986). Learning representation by back-propagation. *Nature*, 323:533–536.

Santamaria, J. C., Sutton, R. S., and Ram, A. (1998). Experiments with reinforcement learning in problems with continuous state and action spaces. *Adaptive Behaviour*, 6(2):163–218.

Smart, W. D. and Kaelbling, L. P. (2000). Practical reinforcement learning in continuous spaces. In *Proceedings of the 17th International Conference on Machine Learning*, pages 903–910.

Strosslin, T. and Gerstner, W. (2006). Reinforcement learning in continuous state and action space. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*.

Sutton, R. S. and Barto, G. B. (1998). *Reinforcement Learning*. MIT Press.

Winfield, A. F. T. and Erbas, M. D. (2011). On embodied memetic evolution and the emergence of behavioural traditions. *Memetic Computing*, 3(4):261–270.