# 16.412 Final Project

# Co-operative Q-Learning

Lars Blackmore and Steve Block
*(This report is by Lars Blackmore)*

## *Abstract*

Q-learning is a method which aims to derive the optimal policy in a world defined by a Markov Decision Process using only the reinforcement signals the learning agent receives.

Recent research has addressed the issue of extending this to the case of multiple agents acting cooperatively; in particular looking at how Q values should be shared between agents to enable cooperation.

An algorithm that has been suggested for this is *expertness based cooperative Q-learning with specialized agents*. Some simulation results have been presented for mobile robots acting in a grid world.

In this project, this cooperation strategy is implemented and tested. A number of implementation issues are investigated and resolved.

This expertness based cooperation method is shown to give an improvement in performance for two distinct cases. The first of these cases is when agents carry out individual learning in separate areas of the state space and then are expected to perform in areas with which they are unfamiliar. The second is when agents explore a similar region of the state space, but have significantly different experience levels.

In the two cases where the algorithm is strong a simpler alternative weighting strategy, Most Expert, is suggested, and shown to be just as effective as the Learning from Experts strategy.

It is also shown that Q-learning in general, and cooperative Q-learning in particular, is largely unaffected by dynamic environments except in specially constructed cases. The *discounted expertness* method is proposed to mitigate the effects of dynamic environments in these cases. Testing shows that this is effective.

Finally, the strength of the above conclusions and their applicability to more general cases are considered.

Code for the project is available online at web.mit.edu/sblock/16.412/project.

*Contents*

## 1. Introduction

This project builds on work on cooperative Q-learning by Ahmadabadi, Eshgh, Asadpour and Tan. This work collectively proposes an algorithm for carrying out Q-learning with multiple cooperative agents by sharing Q values between agents. This sharing process is based on the *expertness* assigned to each agent.

This project aims to analyse the results obtained by these researchers, and if possible to extend both the results and the underlying algorithm for cooperative Q-learning.

## 2. Q-Learning

Q-learning is a form of reinforcement learning which aims to find an optimal policy in a Markov Decision Process world using reinforcement signals received from the world.

A Markov Decision Process is defined as having a set of possible states **S**, a set of possible actions **A**, and a reinforcement function *R(s,a)* based on taking action *a* in state *s*. In addition there is a probabilistic transition function *T(s,a,s')*which defines the probability of transitioning to state *s'* if action *a* is taken in state *s*.

Q-learning defines a value *Q(s,a)* which approximates the lifetime reward for an agent which takes action *a* in state *s* and acts optimally from then onwards. Q values are updated using the Q-learning update equation:

$$Q(s,a) \leftarrow (1-\alpha)Q(s,a) + \alpha\left[r + \gamma \max_{a} Q(s',a)\right]$$

Here, $\alpha$ is the *learning rate* while $\gamma$ is the *discount factor*. Both of these are between 0 and 1, and can be set using various heuristics.

The reinforcement received by the agent is denoted *r*.

Given converged Q(s,a) values, the optimal policy in state *s* is:

$$a = \arg\max Q(s,a)$$

In practice, agents trade off *exploitation* of learned policy and *exploration* of unknown actions in a given state. There is a certain probability that an any given time the action taken will not be that determined by the equation above, but will be a random action. Heuristics to set this probability include a concept of the *temperature* of the system in an analogous manner to simulated annealing.

## *3. Previous Research in Co-operative Q-Learning*

### 3.1 Cooperative Reinforcement Learning

Work by Whitehead[4] and Tan[3] introduced the concept of Q-learning with multiple cooperative agents. Tan suggested a number of different methods by which agents can cooperate. These were the sharing of sensory information, sharing of experiences and sharing of Q-values.

The method used for sharing Q-values was *simple averaging*. In this method, each agent *i* averages each of the other agents' $Q_j(s,a)$ values:

$$Q_i(s,a) \leftarrow \frac{\sum_{j=1}^{n} Q_j(s,a)}{n}$$

Q-learning as described in section 2 is then performed by each agent on the shared values. The sharing step described above can occur at each step of a trial, at each trial, or in fact at any stage in the Q-learning process depending on the exact nature of the implementation.

### 3.2 Expertness Based Cooperative Q-Learning

Ahmadabadi and Asadpour[1] suggested that simple averaging had a number of disadvantages. Firstly, since agents average Q-values from all agents indiscriminately, no particular attention is paid to which of the agents might be more or less suitable to learn from. Secondly, simple averaging reduces the convergence rate of the agents' Q-table in dynamic environments. Eshgh and Ahmadabadi[2] showed that in simulations of robots in a maze world, cooperation using simple averaging gave a significantly lower performance than agents learning independently.

Ahmadabadi and Asadpour[1] proposed a new algorithm called *expertness based cooperative Q learning*[1]. In this method, each agent is assigned an *expertness value* which is intended to be a measure of how expert the agent is. When an agent carries out the cooperation step, it assigns a weighted sum of the other agents' Q-values to be its new value. The weighting is based on how expert one agent is compared to another. In this way, agents consider the Q-values of more expert agents to be more valuable than those of less expert agents.

Ahmadabadi et al.[1,2] suggested a number of methods for assigning expertness to agents. They showed that the most successful of these were based on the reinforcement signals an agent received. They noted that both positive and negative reinforcement contributed to the concept of expertness, and hence proposed, among others, the "*absolute"* measure of expertness which weights positive and negative reinforcement equally.

$$e_i = \sum_t |R_i|$$

It was found that while other expertness measures may be optimal in certain situations, the *"absolute"* measure was the most effective in the general case.

Ahmadabadi and Asadpour[1] also suggested a number of weighting strategies for sharing Q-values. The *"Learning from Experts"* method was shown to be the best of these in terms of the performance of the cooperative agents relative to individual agents.

$$W_{ij} = \begin{cases} 1 & i = j, e_i = e_{\max} \\ 1 - \alpha_i & i = j, e_i \neq e_{\max} \\ \alpha_i \dfrac{e_j - e_i}{\sum_{k=1}^{\infty} (e_k - e_i)} & e_j > e_i \\ 0 & otherwise \end{cases}$$

## 3.3 Cooperative Q-Learning for Specialised Agents

Eshgh and Ahmadabadi[2] extended the expertness based cooperative Q-learning method to include expertness measures for different zones of expertise. In this way, a particular agent could be assigned high expertness in one area, but have a low expertness in another area. A number of different zones of expertness were suggested; *global*, which measures expertness over the whole world, as in reference 1, *local*, where the zones correspond to large sections of the world, and *state* where each agent has an expertness value for every state in the world.

## 3.4 Simulation and Results

Eshgh and Ahmadabadi[2] carried out simulations with mobile robots in a maze world, where the robots received a large reward for moving into the goal state. The robots received a small punishment for each step taken, and also a punishment for colliding with obstacles such as walls. In the following, the terms *agent* and *robot* are used interchangeably.

The maze world was roughly segmented into three sections, with a robot starting from a random location in each section. That is, obstacles were placed in such a way that a robot starting in a certain section, although able to move into a different section, is unlikely to do so during any given trial. Each section of the world had a goal.

Each test started with an *independent learning* phase. During this phase, agents carry out Q-learning as described in section 2 without any form of cooperation. A number of trials are carried out; each trial starts with the agent at some start location, and ends when the agent reaches the goal. Each agent retains its Q-values from one trial to the next, and at the end of this phase, each agent has a different set of Q-values which have been learnt over a number of independent trials.

After the independent learning phase, the agents carry out *cooperative learning*. In this phase, agents carry out Q-learning as before, however the agents share their Q-values using the algorithm described in 3.3. The "absolute" expertness measure and the "learning from experts" weighting strategy were used. The exact timing of this sharing process is not stated explicitly. The expertness zones mentioned in 3.3 were defined so that *global* included the entire world, *local* had three expertness zones, one for each of the world segments, and *state* had an expertness zone for each state in the world.

Simulations showed that the average number of steps to reach the goal reduced by more than 50% when cooperation was used with either *local* or *state* expertness. On the other hand, the average number of steps to reach the goal was *increased* slightly when *global* expertness was used.

### 3.5 Conclusion

In previous research a number of methods by which agents can cooperate by sharing Q-values have been suggested. Of these methods, the *expertness based cooperative Q-learning with specialized agents* method described in 3.3 was shown to give the highest improvement in performance over individual Q-learning for a particular test case (if the *local* or *state* expertness zones were used.)

It was therefore decided that interesting avenues for further research were:

1. Investigate further the benefits and costs of different expertness zone allocations
2. Investigate the performance of the cooperation method in more general test cases (for static worlds)
3. Investigate the effect of dynamic worlds on this cooperation method

These areas are explored in this report.

## *4. Expertness Zones*

Previous work[2] mentions that creating a method to determine expertness zones automatically is a promising area for future research. Some attention was given to this aspect in this project.

It is assumed that while global expertness has been shown to give poor performance[2] compared to state expertness, the latter has additional costs in terms of storing the various expertness levels. Hence there appears to be a trade-off between storage cost and performance benefit.

However to assign different zones to certain sets of states, the 'parent zone' must be stored *for each state*, in general. The memory required for this is linear in the number of states. Hence there is no storage benefit in creating arbitrary expertness zones compared to storing expertness for each zone explicitly as in the *state* expertness zones. Hence in general, state expertness is optimal since it has highest performance and does not cost any more than the other methods in terms of storage.

This conclusion is qualified however by two factors. Firstly, there could be ways of parameterising the zones so that the parent zone for each state does not have to be stored explicitly. The global expertness case is an extreme case of this. Secondly, while it seems intuitively correct, it has not been shown that state expertness gives the best performance in all cases.

Taking the above into account, it was decided to use state expertness for the remainder of the project, and not to spend more time looking at ways to allocate different expertness zones.

## *5. Simulation Setup*

### 5.1 Simulation Format

Simulations were carried out in a number of maze worlds shown in Figure 1, Figure 3, Figure 2 and Figure 4. These maze worlds can have any number of goal states as determined by the individual simulation in progress. Start states are shown by the red squares, while goal states are shown by the blue squares.

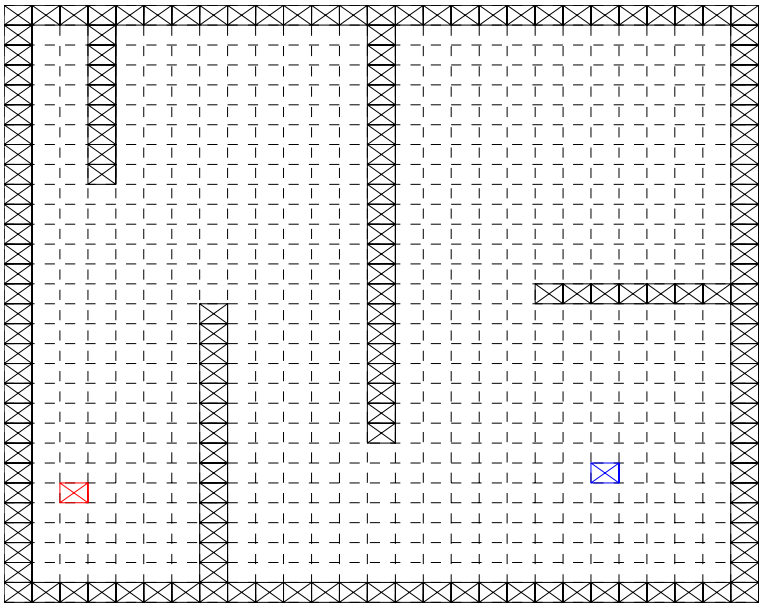

**Figure 1: Simple maze world**

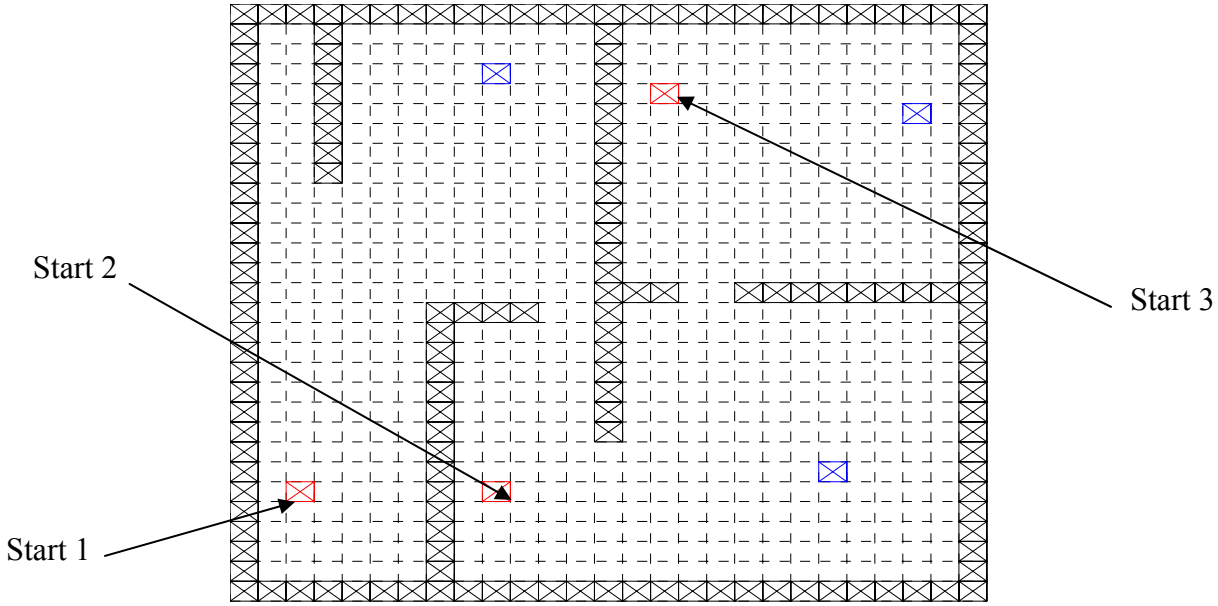


**Figure 2: Segmented maze**

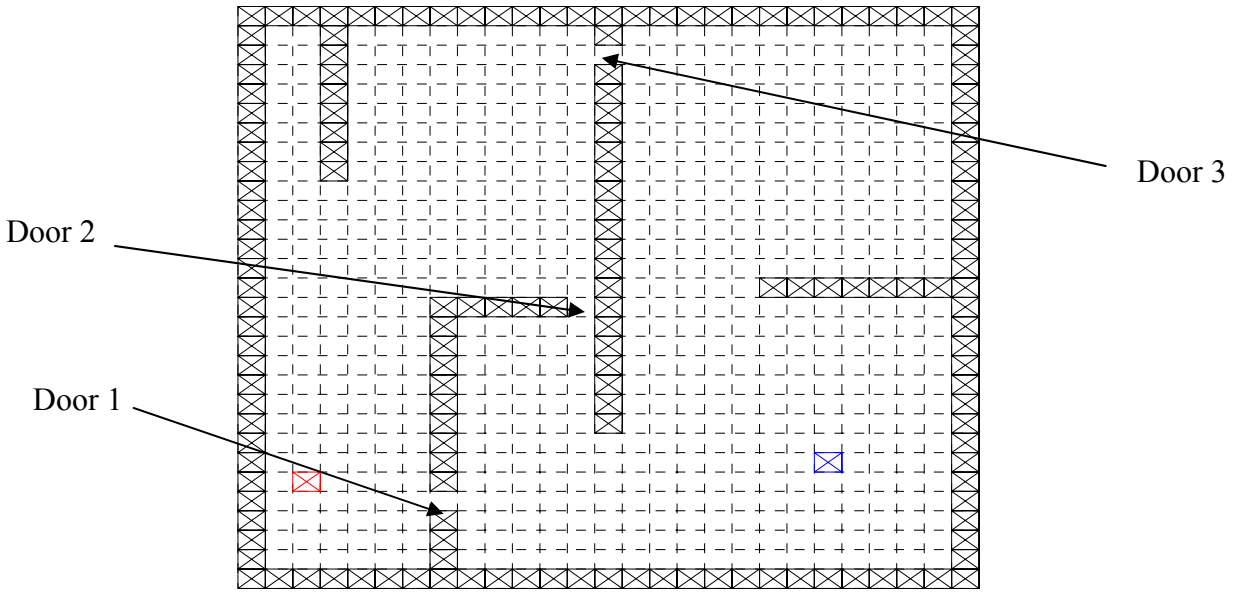


**Figure 3: 'Doors' maze world**

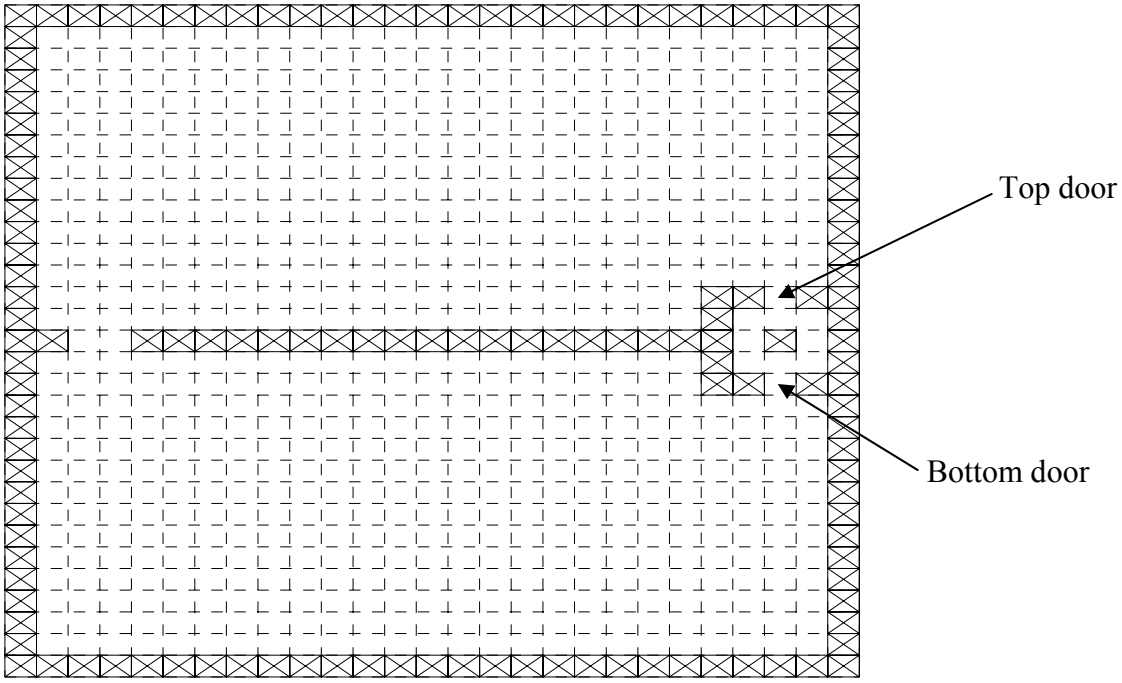


**Figure 4: 'Contrived' maze world**

A simulation consists of two distinct phases:

In the **individual learning** phase, a number of agents carried out a predetermined number of trials. Each agent carries out Q-learning as described in section 2, retaining its Q-table between trials.

Each agent has a fixed start state, and the trial ends when the agent reaches the goal state. Each agent could carry out different numbers of trials, and also start their set of trials at different times.

In the **testing** phase, each agent carries out trials which end when it reaches **any** of the goal states. All the agents start at the same start state; a testing phase is carried out for each agent for each of the start states used in individual learning.

Note that the testing phase is approximately equivalent to the *cooperative learning* phase mentioned in references 1 and 2.

Depending on the nature of a simulation, at any stage, cooperation can occur. At cooperation, the agents share their Q-tables according to the cooperation algorithm being tested. Cooperation does not involve any trials, or indeed any update of the agent's state. In this report the phrases *cooperation* and *sharing Q-values* both describe this event.

For example, a simulation may start with a set of agents carrying out the individual learning phase, followed by a cooperation, where the Q-tables are shared, and finally a testing phase where the agents act using the shared Q tables.

For each of the simulations described in the following sections, this setup is used with certain modifications.

## 5.2 Presentation of Results

This section describes the format used to present simulation results.

*Q-Field Plots*

In order to gain insight into the policy of the agent, pointers were plotted on the grid world in order to show the direction of the action having the maximum Q value in any state. These pointers define the policy of the agent in the world at that instant.

In order to represent the magnitude of the maximum Q value in any given state, different magnitudes were assigned different colours in the order of red, green, blue and black in decreasing order. Hence red regions correspond to those where the reward at the goal has been propagated through successive trials. Black regions correspond to Q values which are close to zero, usually indicating that the state has not been visited.

Example of Q-field plots are shown in Figure 5.

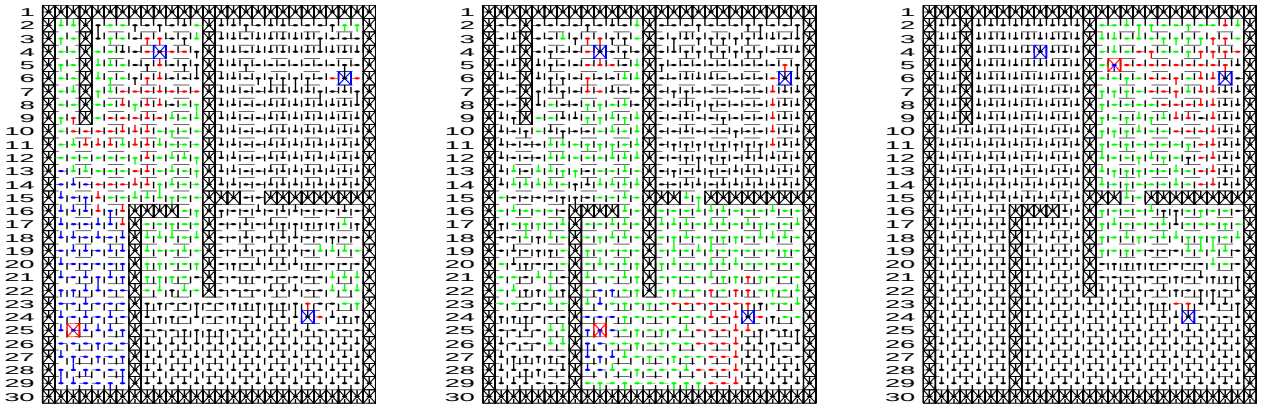

**Figure 5: Example Q-field plots**

*Performance Plots*

The performance of an agent is measured in the number of steps taken to reach the goal in any given trial. Both individual and testing phase results are plotted on a single graph to allow for direct comparison. The x-axis of the graph is time, where an increment of time is one trial. At time *t=0*, the individual phase ends and the testing phase begins; usually at this point cooperation will occur (sharing Q tables). Different coloured lines represent different agents, and in some cases points will be used instead of lines for clarity.

An example of a performance plot is shown in Figure 6.

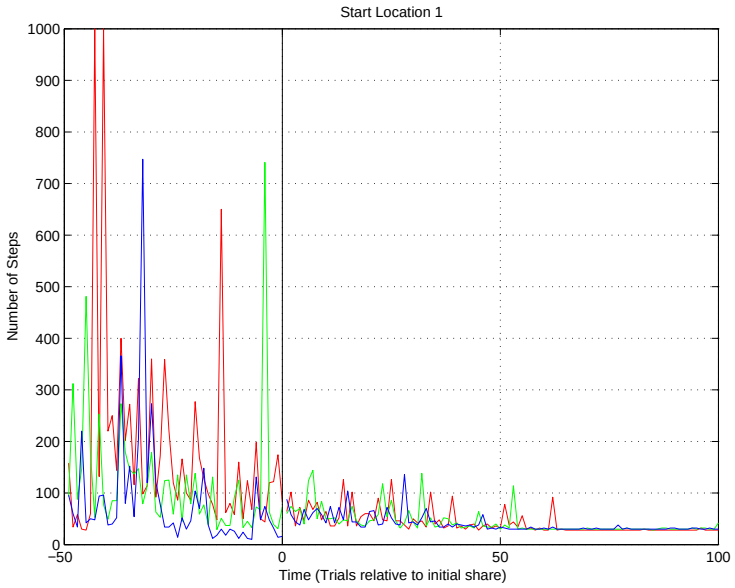


**Figure 6: Example performance plot**

## *6. Initial Findings*

During initial implementation, a number of issues were discovered and addressed. These are described in this section.

### 6.1 Initial Q-Values

Before starting the Q-learning process, the Q(s,a) values must be initialised. Ahmadabadi et al. assigned random values between the maximum and minimum reinforcement signals in the world. While this is one common approach, another common approach in Q-Learning is to assign zeros to all Q values initially.

Tests showed that the difference in performance between the two was small, with random values reducing overall performance and convergence rates slightly as might be expected. Most importantly, however, initialising the values to zero allowed far greater insight into the Q-learning process. In particular, it enabled the Q-fields described in section 5.2 and analysed in subsequent sections to be plotted. Hence Q values were initialised to zero throughout the project.

### 6.2 Q-Learning Parameters

Eshgh and Ahmadabadi used Q-learning parameters of learning rate $\alpha=0.8$ and discount factor $\gamma=0.9$. The temperature used in the action selection model was $T=1.5$ and the impressibility factor for all agents $\alpha_i=0.8$. In the simulations presented in this project, the same learning rate and discount factor were used; however different temperature and impressibility factor values were used.

The action selection temperature determines the likelihood that an action will be selected which is not the current estimate of the optimal policy; i.e. an action which does not maximise $Q(s,a)$. The temperature value is a representation of the trade-off between exploitation and exploration. For the experiments which follow, the temperature was set to $T=0$ so that at all times the estimated optimal policy was followed. This was done to reduce noise in the results, to assist analysis of the underlying process, and in addition was not found to yield different conclusions in the examples tested.

The impressibility factor $\alpha_i$ is the proportion by which agent $i$ weights the other agents' Q values during cooperation. If $\alpha_i$ is less than one, then an agent will incorporate its own Q value regardless of how little expertness it has. This was found to cause problems in dynamic environments with discounted expertness since an agent may have large Q-values, but low expertness since the experience is old. In the $\alpha_i<1$ case, the agent will still have a contribution from its own, wrong, Q-values, which goes against the idea of discounted expertness. Hence $\alpha_i=1$ was used throughout.

## 6.3 Start Locations

Eshgh and Ahmadabadi used random start locations for both the *individual learning* and the *cooperative learning* phases. This slows the convergence of the agents' Q tables and also means that in order to evaluate performance effectively, the deviation from the optimum path would ideally be calculated for each path taken. Since the optimum path length is a function of the start location, this is not simple. In addition, initial testing showed that no additional insight is gained from the use of random start locations. Hence predetermined start locations were used throughout.

## 6.4 Learning during Test Phases

It is not clear in references 1 and 2 whether during the *cooperative learning* phase, Q-learning is being carried out (despite the name of the phase). Whether or not to carry out Q-learning during the *testing phase* in this project was therefore a point for consideration.

It was found that carrying out the testing phase without Q-learning is highly unsuccessful, for the following reason:

An individual agent carrying out Q-learning in a grid world is guaranteed not to get stuck in infinite loops because the Q-values corresponding to the (s,a) pairs in the loop get updated with the path cost of the agent's motion. Hence at some point in time, alternative actions will be chosen since they now maximise Q(s,a), taking the agent out of the loop.

It was found that after cooperation, because Q values have been assigned in a discontinuous manner from a number of different agents, it is likely that there will be many such loops in the Q field. Without Q-learning in the testing phase, the agent can become stuck in these loops, relying on randomness in the motion model to escape (usually only to be stuck in another loop immediately). Testing showed that this did indeed happen, and performance after cooperation was dismal. Hence it was concluded that Q-learning *must* continue during the testing phase.

## 6.5 Repeated Cooperation

An open question was whether or not agents should cooperate by sharing their Q-values at regular intervals during the testing phase or just once at the start. Initial results showed that there was no performance improvement to be had by sharing at regular intervals. Results presented later in the project show that cooperation has little effect when the overall expertise of each agent is similar, and that after cooperation all agents have similar overall expertise. Hence sharing after the initial cooperation will not improve performance. On the other hand sharing takes approximately 10 times longer than an individual trial, significantly increasing the computer time needed for simulations. As a result, all further simulations in the project were carried out using a single share of Q values between the individual phase and the testing phase.

## 6.6 Assessment of Performance

One of the main aims of this work is to assess the performance of cooperating agents to agents which have only carried out individual learning. There are two possible options here with regard to mobile robots seeking a goal:

1. Compare how long it takes for the individual agents as a group to find the goal compared to how long it takes the cooperative agents as a group.
2. Compare how the performance of any particular agent is improved by cooperation with the other agents.

In the first option, the time for the most capable agent to find the goal is the measure of performance. It can be seen that by looking at the performance of each agent with and without cooperation (option 2), the most capable agent's performance can be assessed and hence performance based on option 1 can also be assessed.

As a result, the performance of each agent using individual learning only and after cooperation is compared in this project, since it encompasses the group performance measure mentioned above.

## *7. Performance in Static Worlds*

In this section, the cooperation algorithm described in 3.3 is implemented with *state* expertness zones and tested in a maze world with multiple agents.

The aims are to:

1. Duplicate the results found in previous research
2. Assess the performance of the algorithm in more general cases than those tested by Ahmadabadi et al.

### 7.1 Simulation 1: Improvement in Performance over Non-Cooperative Agents in a Segmented World

It was desired to replicate Ahmadabadi's result that in a segmented world, cooperating agents using the algorithm described in 3.3 have better performance than agents which do not cooperate. In order to do this, simulations were carried out with the segmented maze shown in Figure 2. It can be seen that the world is roughly partitioned into three segments, and that an agent starting at one of the start states is likely to remain in the corresponding world segment for the duration of a trial. The goal states are shown in blue and the start states are shown in red.

For this simulation, three agents were used. Individual learning was carried out with agent 1 starting at start state 1, agent 2 starting at start state 2 and agent 3 starting at start state 3. For any particular agent, the trial ended when that agent reached any goal.

After individual learning, the agents shared their Q-tables using the algorithm described in 3.3 with *state* expertness zones. Then the test phase was carried out for the cases where all agents start at either start state 1, 2 or 3. Each trial ended for an agent when it reached any goal. Note that at the start of the test phase for any given start locations, the initial Q-fields (obtained immediately after sharing) are the same as at the start of the test phase for any other start location. In other words, to compare the different testing phases fairly, the Q-tables at the start of each test phase is re-initialised.

During the individual phase, each agent carried out 50 trials. During the test phase, each agent carried out 100 trials for each of the three start locations.

In order to compare the case of cooperation to the case of individual learning alone, a test phase was carried out starting without any sharing of the Q-tables. This simulation represents what would happen in the case of individual learning only, and is denoted simulation 1a. The simulation *with* cooperation is denoted simulation 1b.

The results from simulation 1 are shown from Figure 7 to Figure 12.

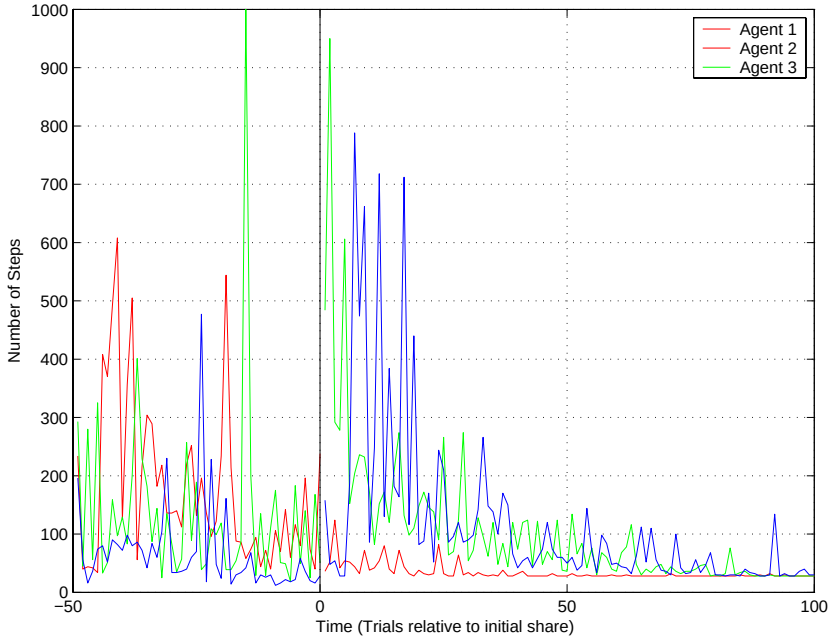


**Figure 7: Simulation 1a (no cooperation) performance plot for start location 1**

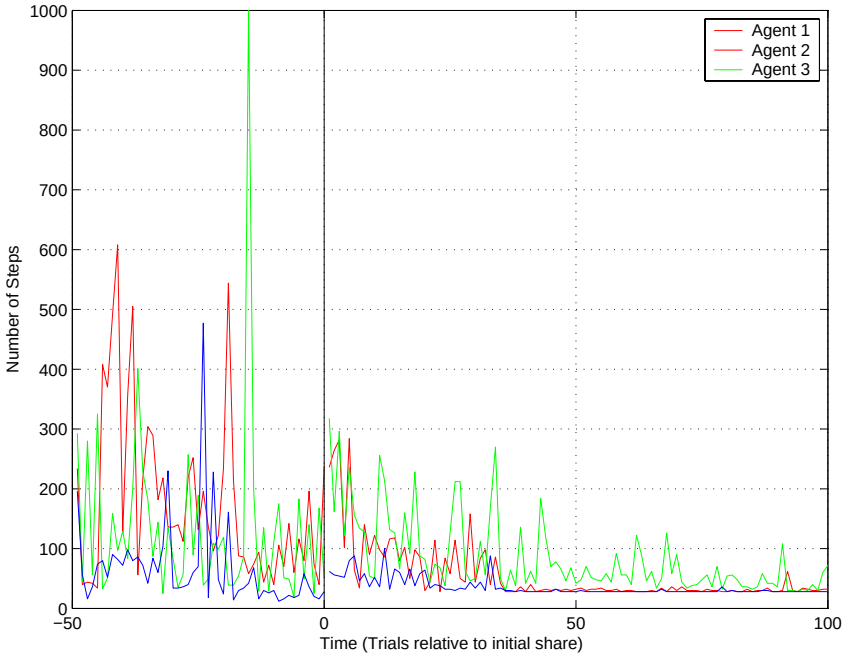


**Figure 8: Simulation 1b (cooperation) performance plot for start location 1**

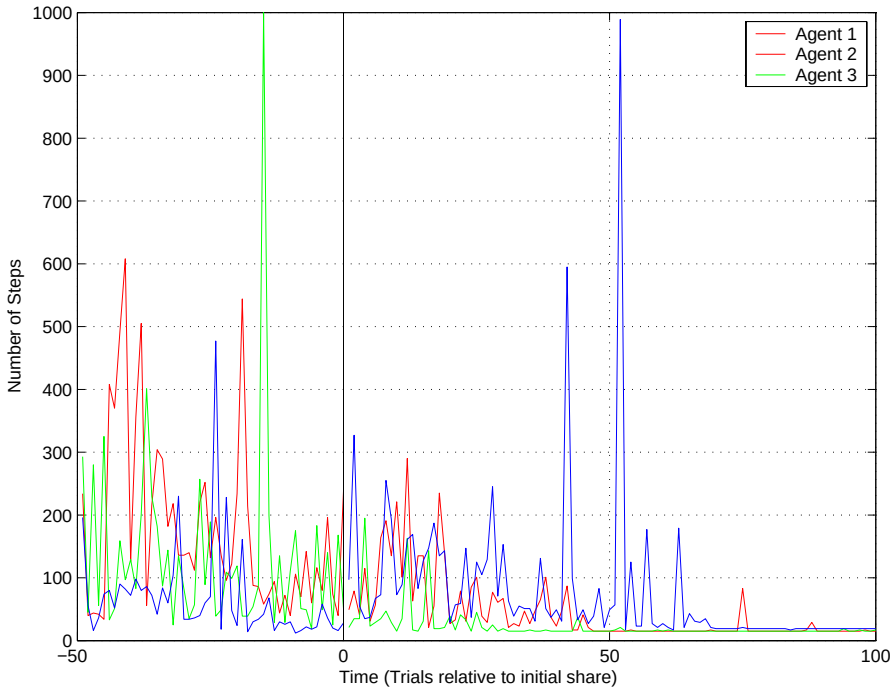


**Figure 9: Simulation 1a (no cooperation) performance plot for start location 2**

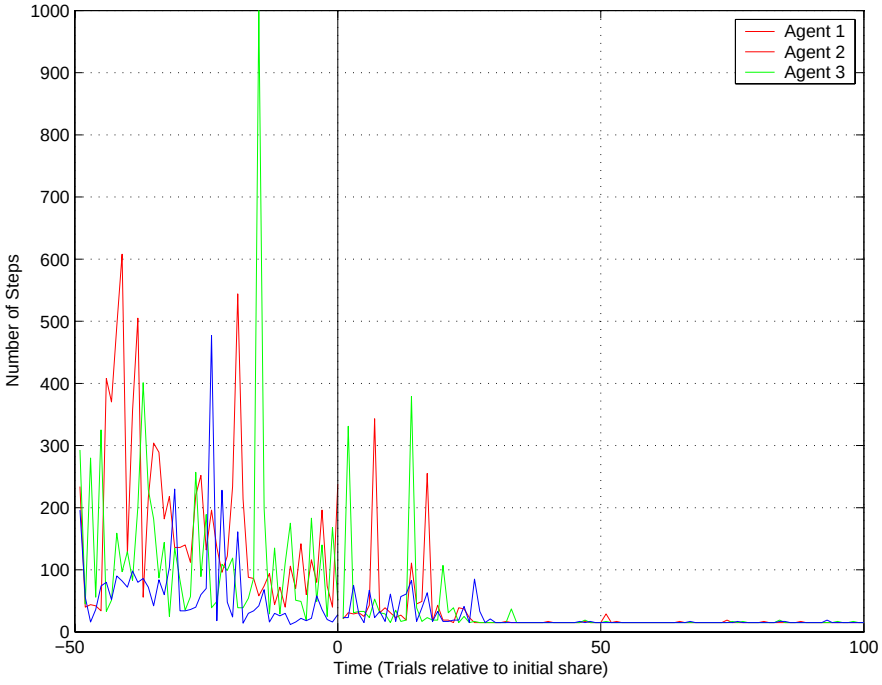


**Figure 10: Simulation 1b (with cooperation) performance plot for start location 2**

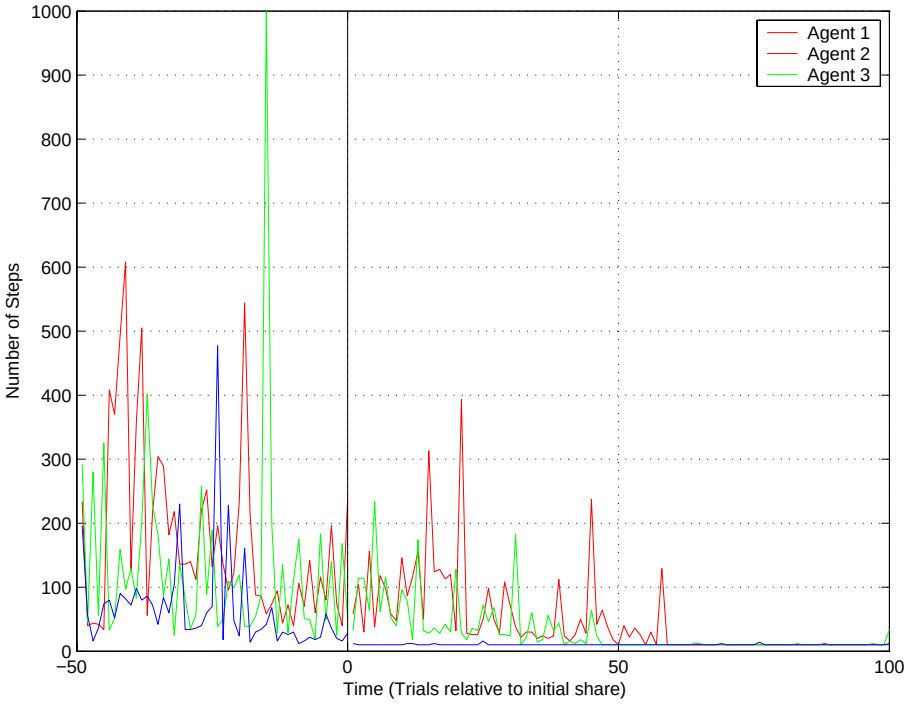


**Figure 11: Simulation 1a (no cooperation) performance plot for start location 3**

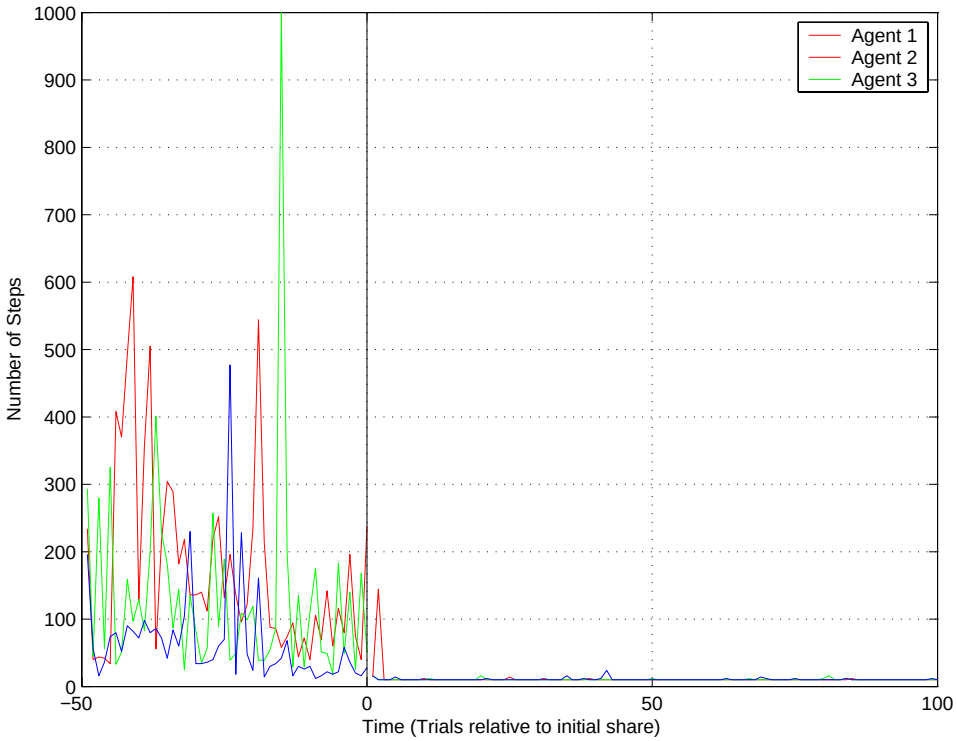


**Figure 12: Simulation 1b (with cooperation) performance plot for start location 3**

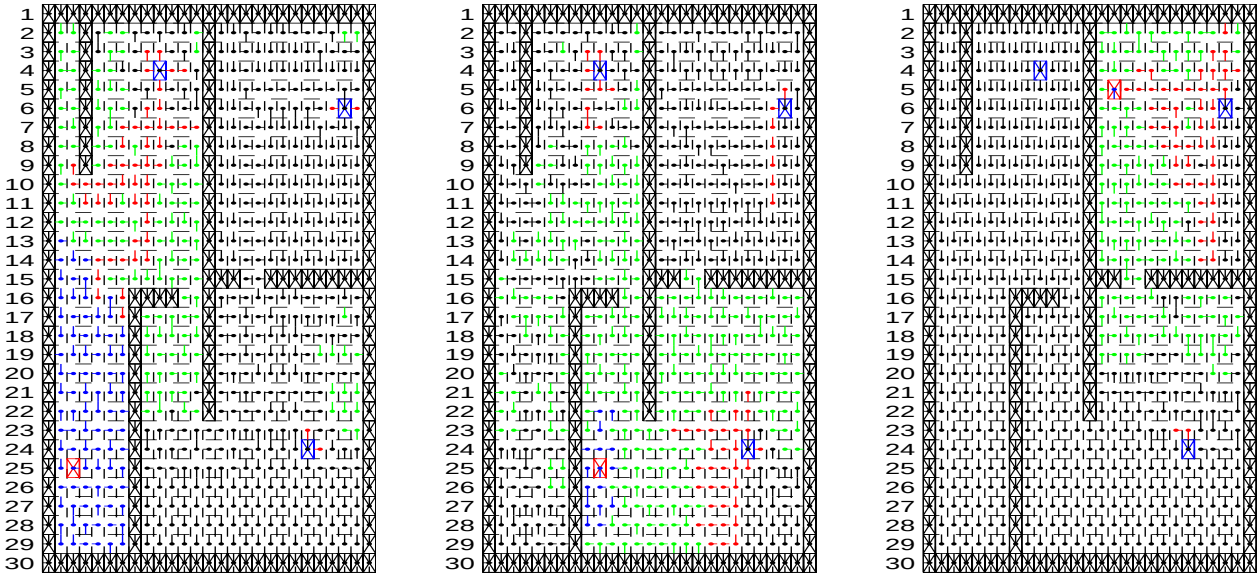

**Figure 13: Simulation 1a, 1b Q-Field at end of individual phase**

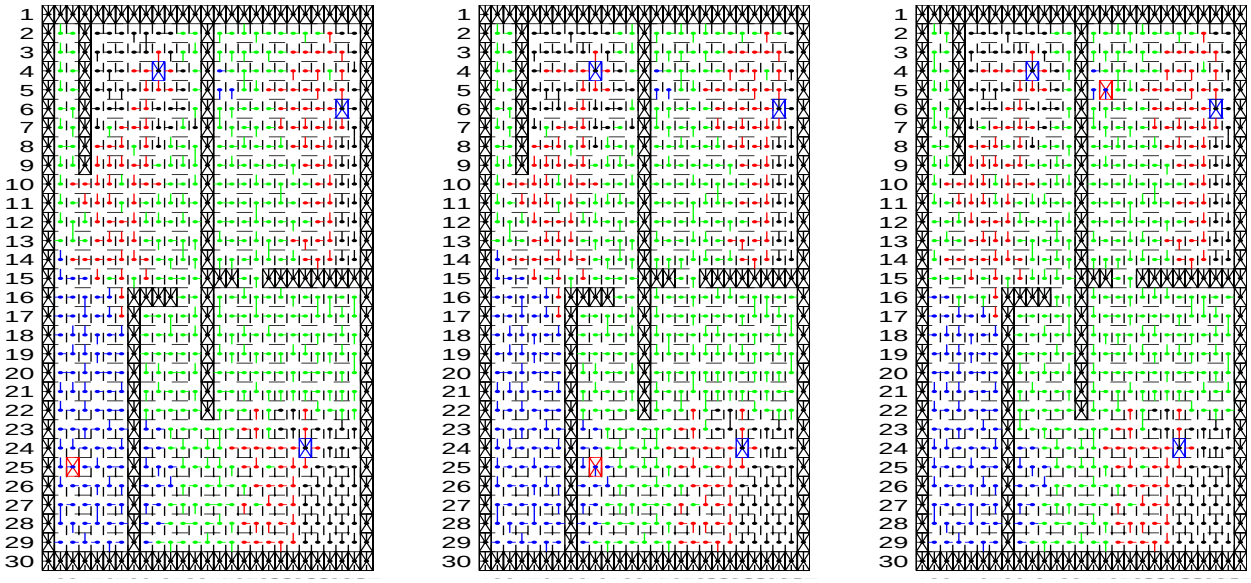

**Figure 14: Simulation 1b Q-field after sharing**

*Simulation 1: Discussion*

For both simulation 1a and simulation 1b, it can be seen that for the individual learning phase from *t=-50* to *t=0*, all 3 agents' performance converges over time, showing that the agents have learnt a relatively good policy in their respective world segment.

For the case where no cooperation is carried out (simulation 1a), it can be seen that at the beginning of the test phase, the performance of one agent continues to converge, while the performance of the other two is significantly worse than during the individual phase. This is

because one agent has learnt a policy relevant for the start location in question, but the other two have almost no experience in this segment of the world. During the test phase, however, the agents continue to learn individually and all three policies eventually converge.

For the case where the agents share their Q tables at *t=0* (simulation 1b), it can be seen that the performance of all three agents is extremely good during the test phase. By sharing, an agent which would otherwise have no learned policy in the segment in which it is being tested can have Q-values which represent a relatively converged policy, and continue to make that policy converge by continued learning.

The effectiveness of sharing Q-values can be seen in Figure 13 and Figure 14. At the end of the individual learning phase, before sharing (Figure 13), each agent has somewhat converged Q-values only in its respective world segment. In other segments, an agent may have no knowledge whatsoever about the location of the goal (represented by all black pointers). After sharing (Figure 14), all the agents have somewhat converged Q-values in all of the areas relevant to all of the goals and from any start location.

It has been shown, therefore, that in this segmented grid-world, sharing Q-values in the manner described in 3.3 improves the performance of the agents, in that agents which have no *individual* experience of a region of the world gain good Q-values for that region; these Q-values are used to determine a close-to-optimal policy for finding the goal.

As one would expect, however, the performance of the agent which *does* have experience in the world segment in which the test is being conducted does *not* improve. Hence Ahmadabadi's result has been confirmed.

## 7.2 Simulation 2: Equal Experience Agents in a General Maze

It was desired to test the Q-sharing algorithm in more general worlds. Simulation 2 was carried out in the simple maze world shown in Figure 1. Three agents were used, all starting from the location shown on the map and with a single goal, also shown. The simulation consisted of an individual phase, with each agent carrying out 50 trials, followed by cooperation when the agents shared their Q-tables, followed by 100 trials in the test phase.

Since each agent carries out the same number of trials from the same start location, they have roughly equal experience in the world.

For comparison, as in simulation 1, a test phase was carried out without any sharing of Q-values, but with each agent having the same Q-value it had obtained after its individual phase. This simulation is denoted simulation 2a, and the simulation *with* cooperation is denoted simulation 2b.

The results for simulation 2 are shown from Figure 15 to Figure 18. Note that there is only one start location in this simulation.

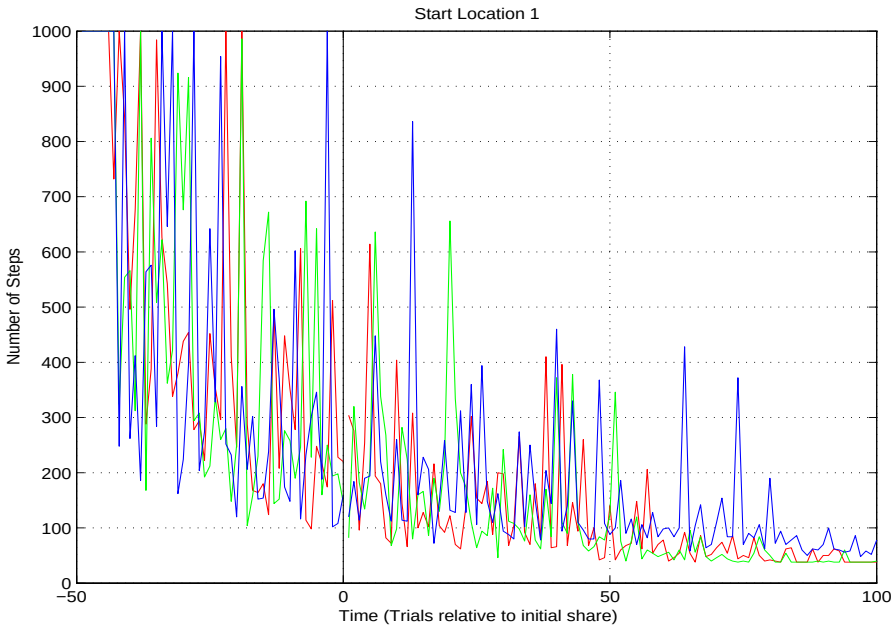


**Figure 15: Simulation 2a (no cooperation) performance plot**

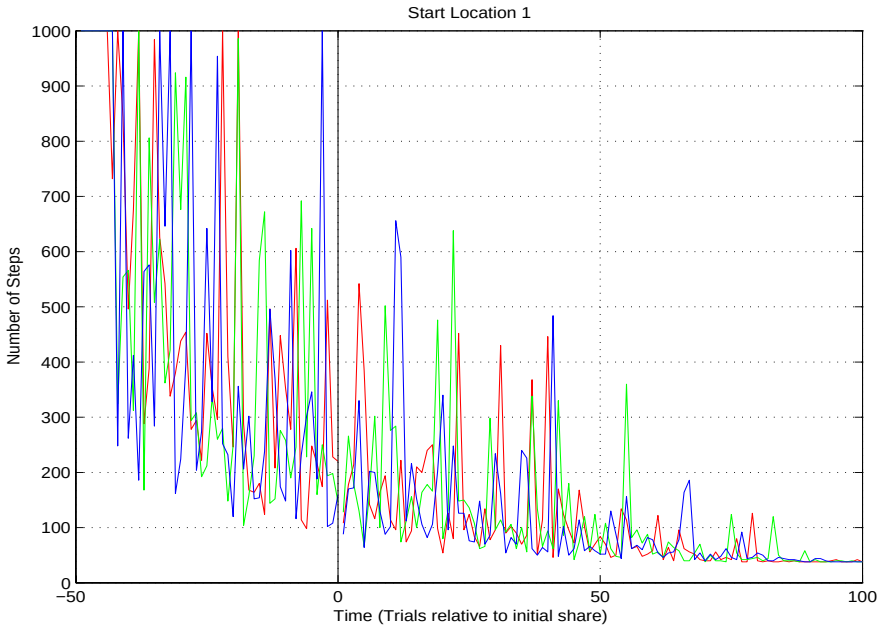


**Figure 16: Simulation 2b (cooperation) performance plot**

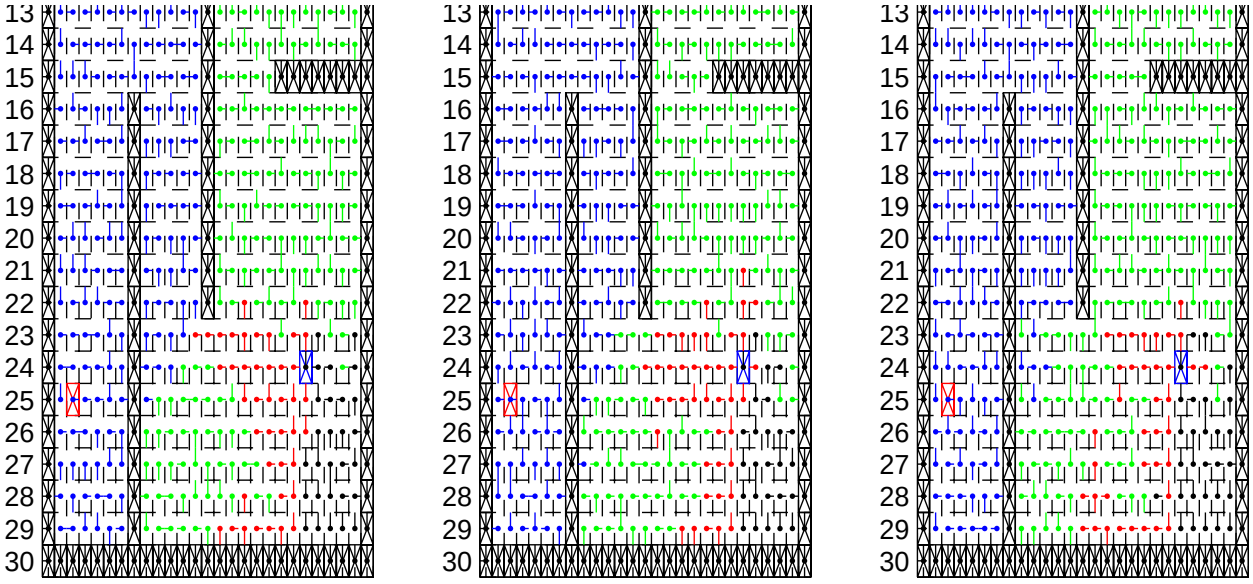

**Figure 17: Simulation 2a and 2b Q-field after individual phase**

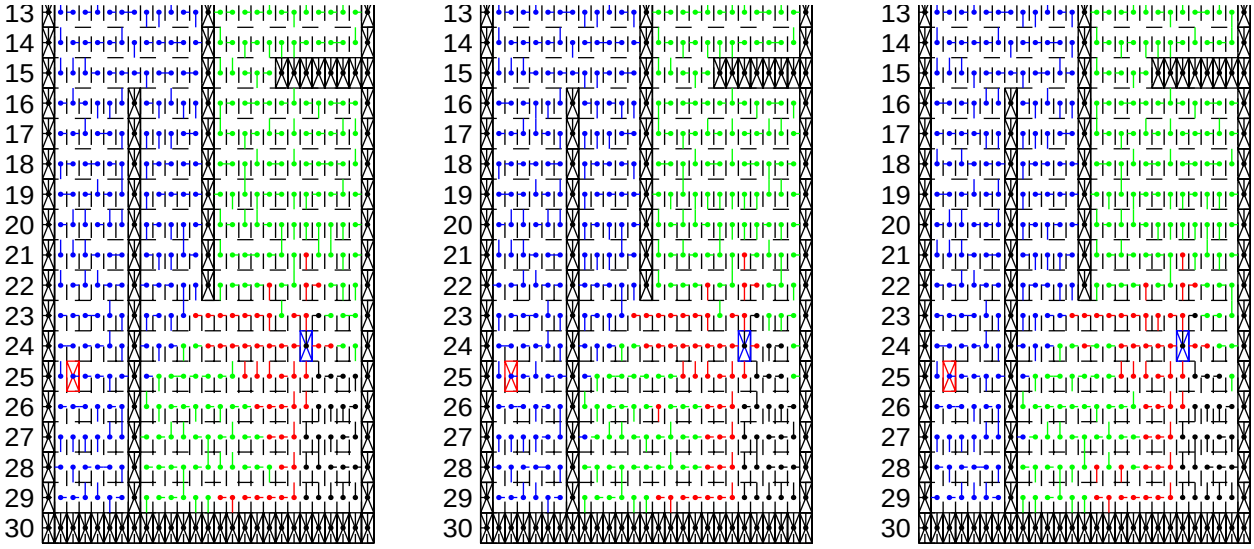

**Figure 18: Simulation 2b Q-field after sharing**

*Simulation 2: Discussion*

The performance plots show that cooperation has not improved the performance of the agents in comparison to the agents using individual learning only. The policies of all three agents continue to converge after cooperation, however cooperation does not affect the performance or the rate of convergence in any noticeable way.

Figure 17 shows the Q-fields after the individual learning phase. It can be seen that the regions of colour are very similar in all three agents, indicating that the various magnitudes of Q-values are distributed similarly across the different agents. This means that after sharing, the Q-fields will

be largely similar to the Q-fields before sharing; it can be seen that this is indeed the case in Figure 18.

In general, the number of steps taken for an agent to find the goal is related to how far the converged policy, in red, has propagated from the goal towards the start. If it has propagated all the way to the start, the path the agent takes will be optimal (ignoring the non-determinism in the world).

It can be seen therefore, that sharing Q-values in this case will have no significant impact on the performance of the agent (measured by the number of steps from the start to the goal), as was observed in the simulation.

## 7.3 Simulation 3: Different Experience Agents in a General Maze

Cooperation did not provide any advantage in the general case for agents with roughly equal experiences. Simulation 3 attempts to determine whether cooperation can be beneficial when agents have different levels of experience in the world.

Simulation 3 is set up in exactly the same manner as simulation 2, with the simple maze world shown in Figure 1. The only difference is that the agents 1, 2 and 3 carry out 100, 50 and 10 individual trials respectively.

As before, simulation 3a shows the case without cooperation, and simulation 3b shows the case with cooperation (sharing Q values at $t=0$). The results are shown from Figure 19 to Figure 22.

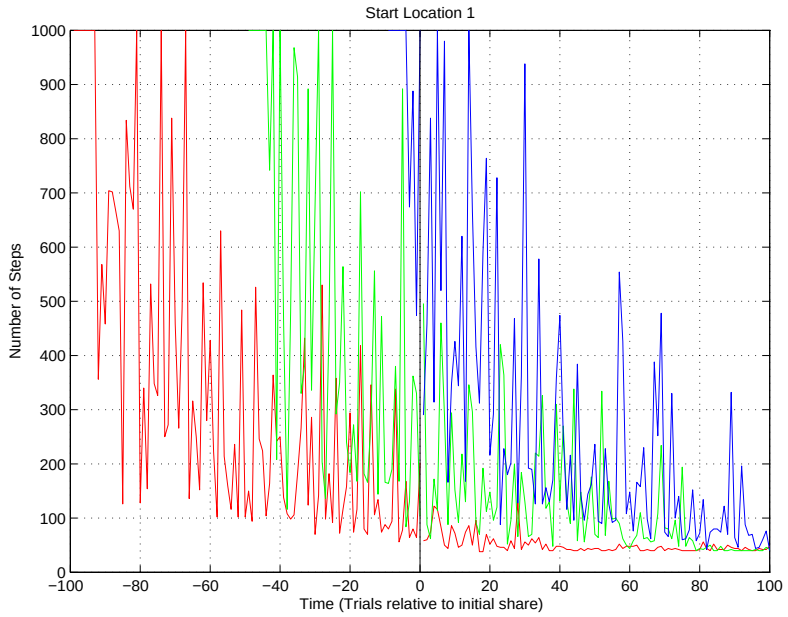


**Figure 19: Simulation 3a (without cooperation) performance results**

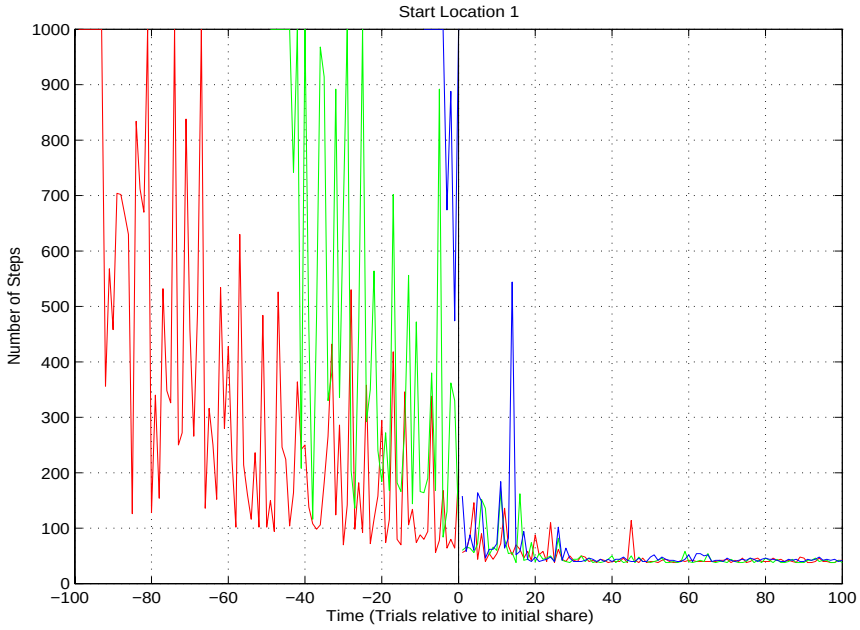


**Figure 20: Simulation 3b (with cooperation) performance results**

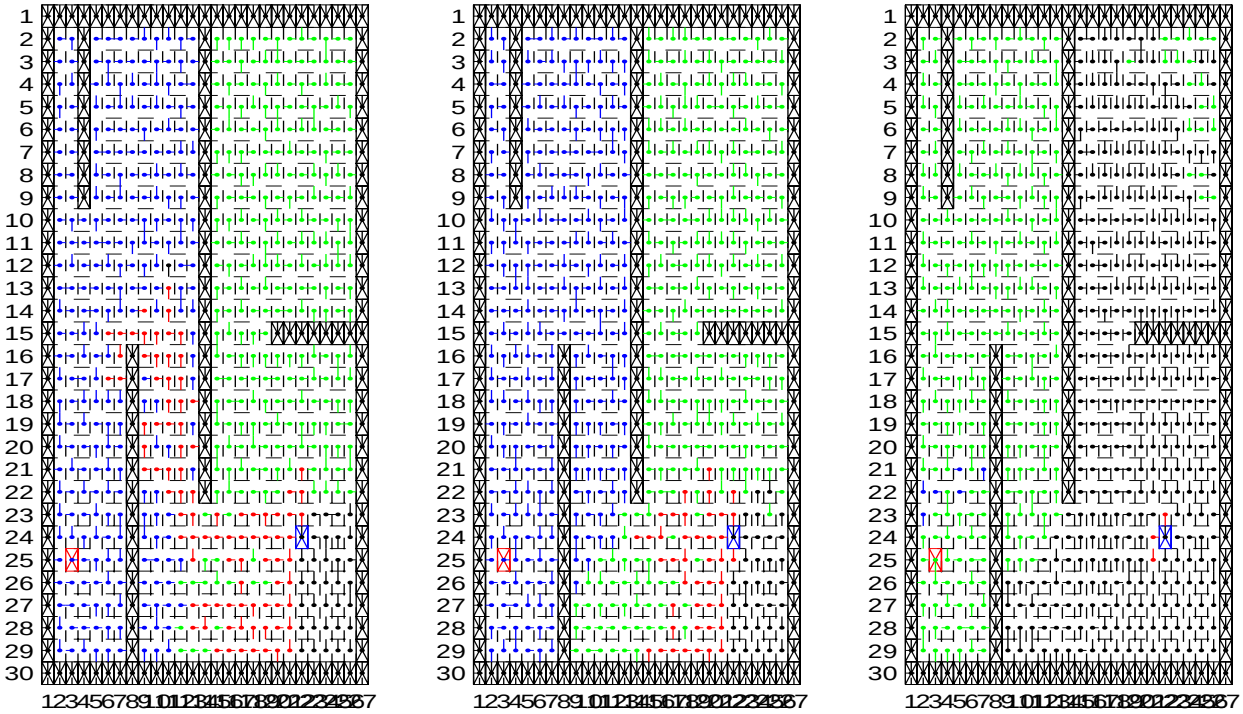

**Figure 21: Simulation 3a and 3b Q field after individual trials**

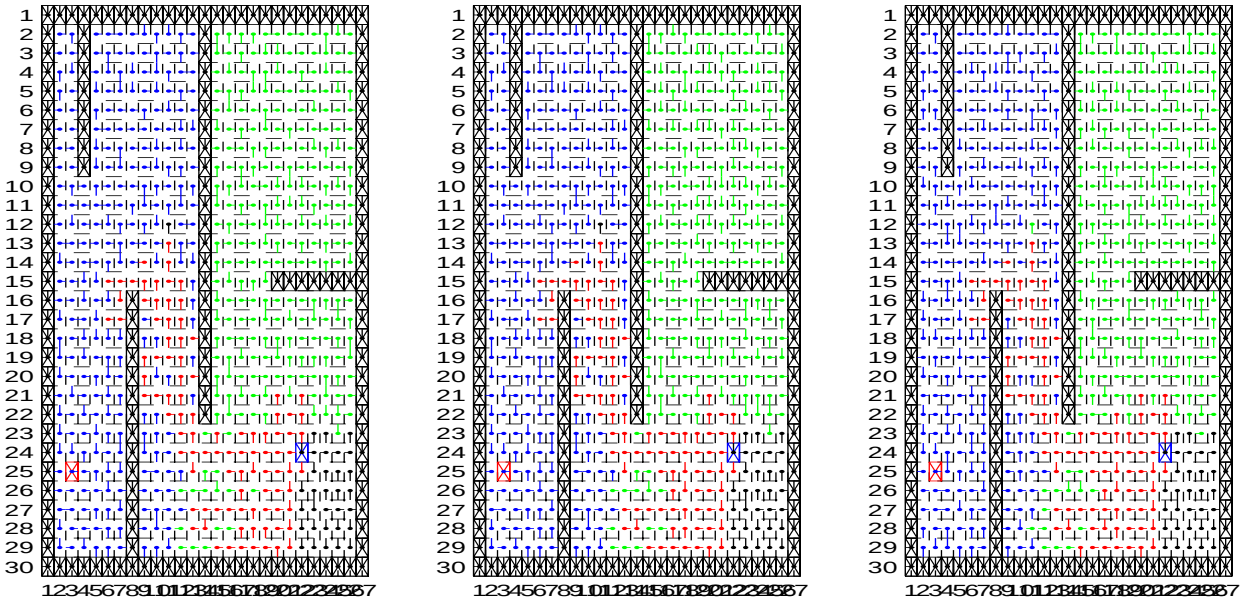

**Figure 22: Simulation 3b Q field after sharing**

Figure 19 shows that without cooperation, all agents continue to converge to the optimal policy independently of one another. With cooperation, as in Figure 20, it can be seen that at *t=0* when the Q tables are shared the performance of the two less experienced agents increases dramatically. In fact the performance instantly becomes very similar to that of the most experienced agent. After the share, the policies continue to converge to the optimum.

It can also be noted that the performance of the most experienced agent (in red) is *not* improved. In fact, according to the Learn from Experts weighting system described in section 3.2, the most expert agent will assign all other agents' Q values a zero weighting, ignoring them completely. In this case, the most experienced agent will almost always have a greater expertness value in any given state, and hence its Q table will be largely unchanged by the share.

Figure 21 and Figure 22 show the Q fields before and after sharing. It can be seen that the red regions denoting converged Q values are small in agents 2 and 3 but large in agent 1. As noted previously, the extent to which these converged values approach the start state is a large factor in the performance of the agent in finding the goal. After cooperation, the regions of converged Q-values have increased greatly in agents 2 and 3, becoming as large as that of agent 1. Hence it would be expected that the performance of agents 2 and 3 would be greatly increased after sharing the Q-values as observed in the simulation.

## *8. Learning from the Most Expert Agent Only*

In section 7 Ahmadabadi et al's results were confirmed by showing that cooperation is better than individual learning in segmented environments, and in a general maze world only if the experience levels of the agents are significantly different.

In both of these cases, in any given state the expertness of one agent will be significantly higher than all the other agents. This means that to a close approximation, all of the less expert agents will acquire the Q value of the most expert agent, while the Q value of the most expert agent will not change. Hence the choice of the weighting mechanism is largely irrelevant, since all agents simply take the Q value from the most expert agent. Hence a simpler weighting strategy, known as *most expert* is proposed and is defined by the following weighting:

$$W_{i,j} = \begin{cases} 1 & e_j = e_{\max} \\ 0 & otherwise \end{cases}$$

This strategy was tested as an alternative to the Learning from Experts strategy presented by Ahmadabadi et. al. The results are shown in Figure 23 and Figure 24.

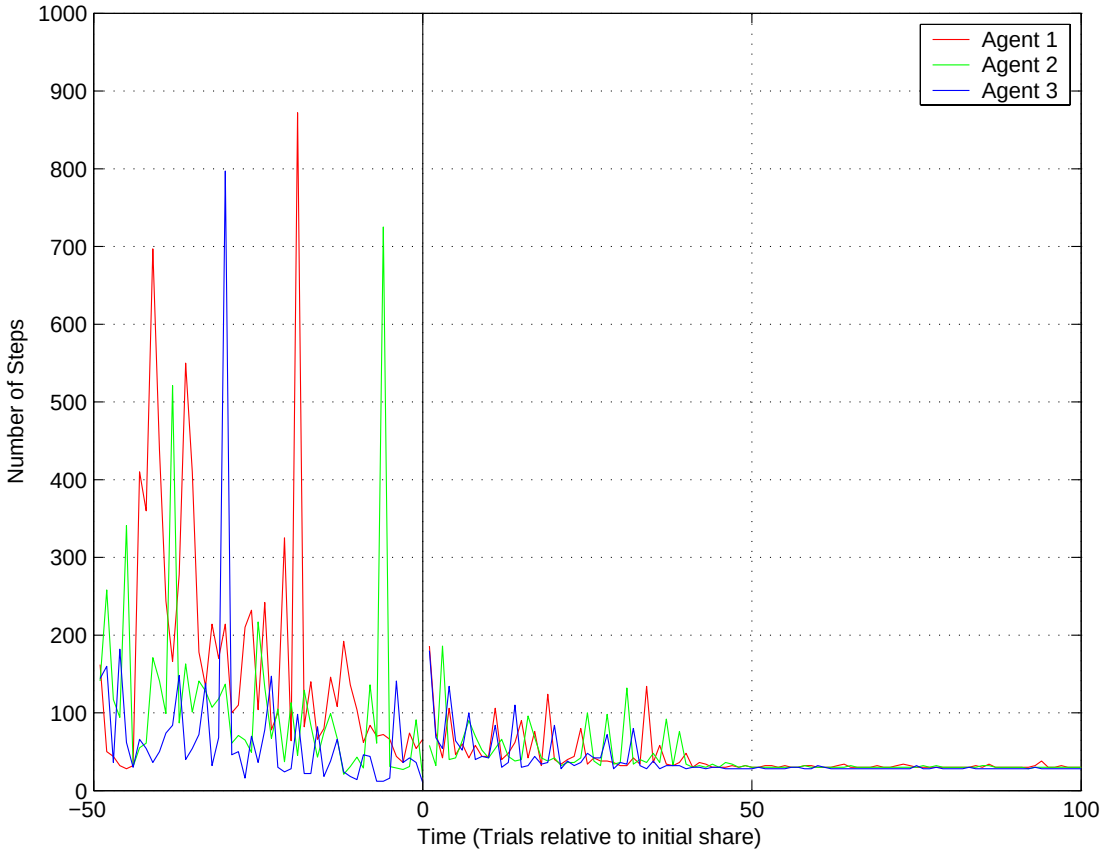


**Figure 23: Performance plot in segmented world using Learning from Experts weighting**

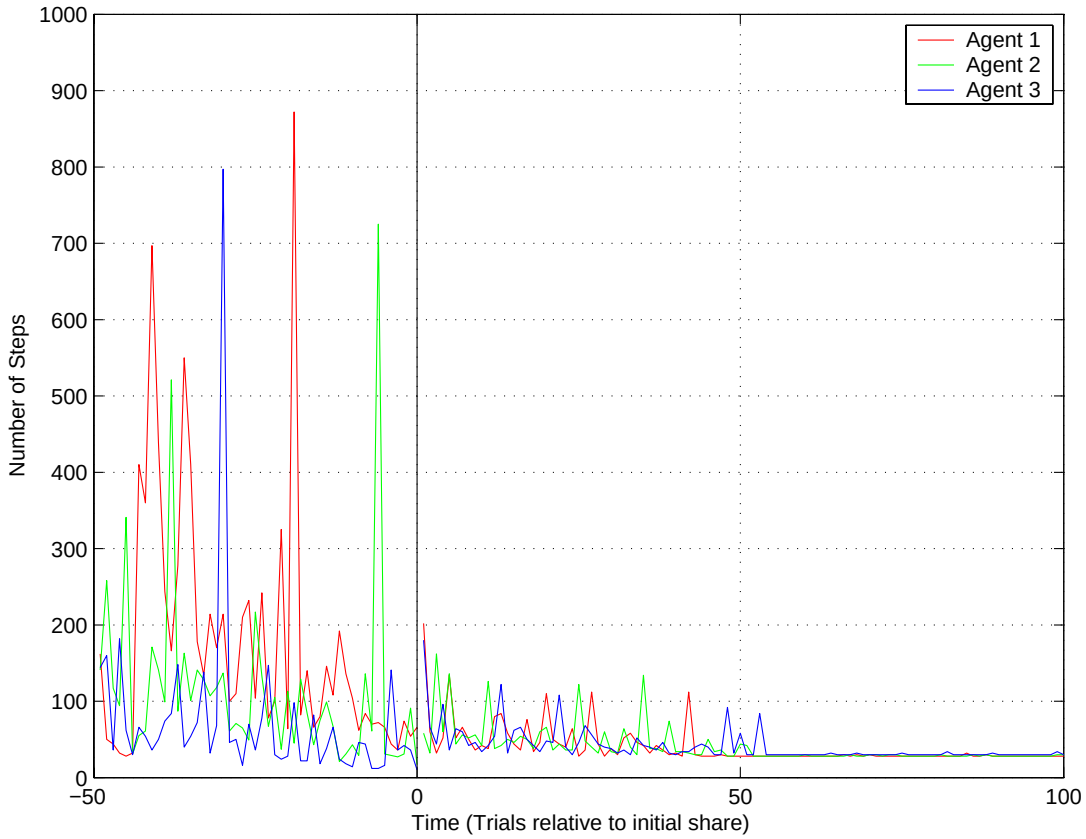


**Figure 24: Performance plot in segmented world using Most Expert weighting**

These performance plots show that very similar performances were obtained for both Learning from Experts and Most Expert weighting strategies, as expected. In addition the Q-fields after sharing shown in Figure 23Figure 25 and Figure 26 are very similar for both methods.

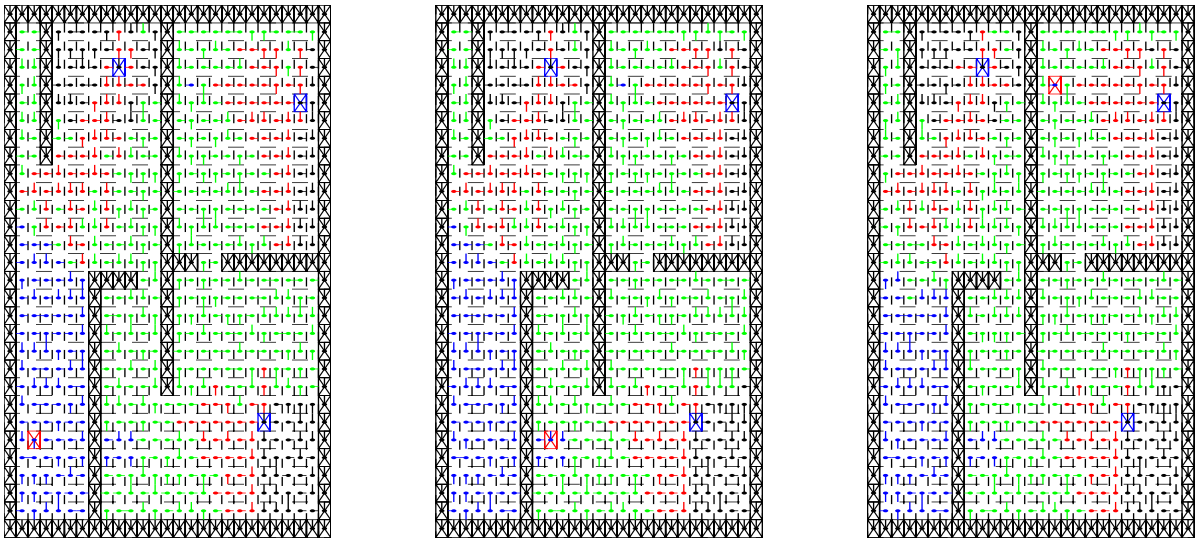

**Figure 25: Q-field after sharing for Learning from Experts method**

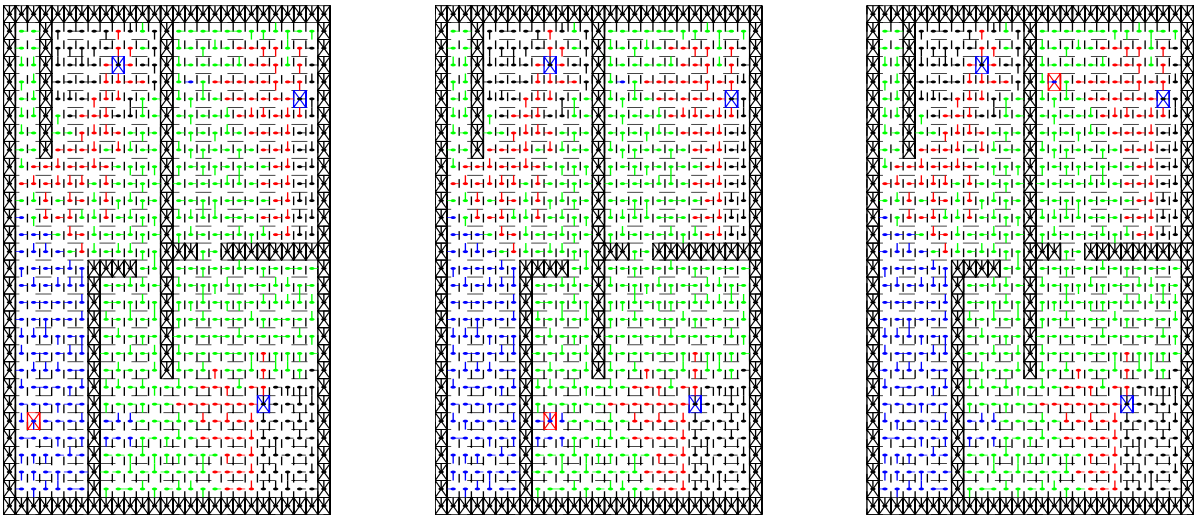

**Figure 26: Q-field after sharing for Most Expert method**

This method does, however, lead to Q tables which are homogeneous across all agents. Ahmadabadi et. al. note that homogeneous policies limit the ability of the group of agents to adapt to changes in the world. For this reason, and for continuity with the earlier results of the project, it was decided to continue with the cooperation algorithm involving the Learning from Experts weighting strategy.

## *9. Performance of Cooperation Algorithm in Dynamic Worlds*

It was mentioned in section 3.5 that the effect of dynamic worlds on the cooperation algorithm would be investigated. This section describes a number of simulations used to do this and the results gained from those simulations.

### 9.1 Simulation 4: Performance in a General Dynamic Maze

For simulation 4 the world was made dynamic by creating a finite probability that at any given timestep an obstacle would appear where there had previously been an unoccupied cell. Once obstacles appeared, they did not disappear. The probability could be set to adjust the rate at which the world changed.

The simple maze shown in Figure 1 was used for this simulation. Three agents carried out an individual learning phase with 50 trials each, followed by sharing of the Q values, followed by a test phase with each agent carrying out 100 trials each. There was one start and one goal as shown in Figure 1.

Simulation 4a was carried out with the probability of an obstacle appearing being such that on average 1 obstacle would appear each trial.

In Simulation 4b this probability was set so that on average one obstacle would appear every 10 trials.

The results for these simulations are shown in Figure 27 and Figure 28.

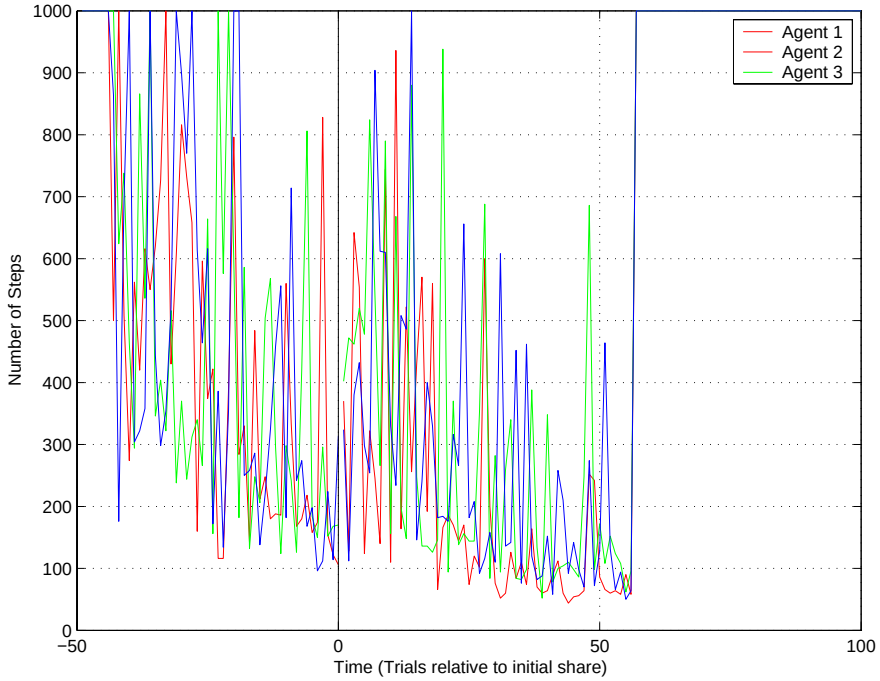


**Figure 27: Simulation 4a (general dynamic maze, p=1) performance plot**

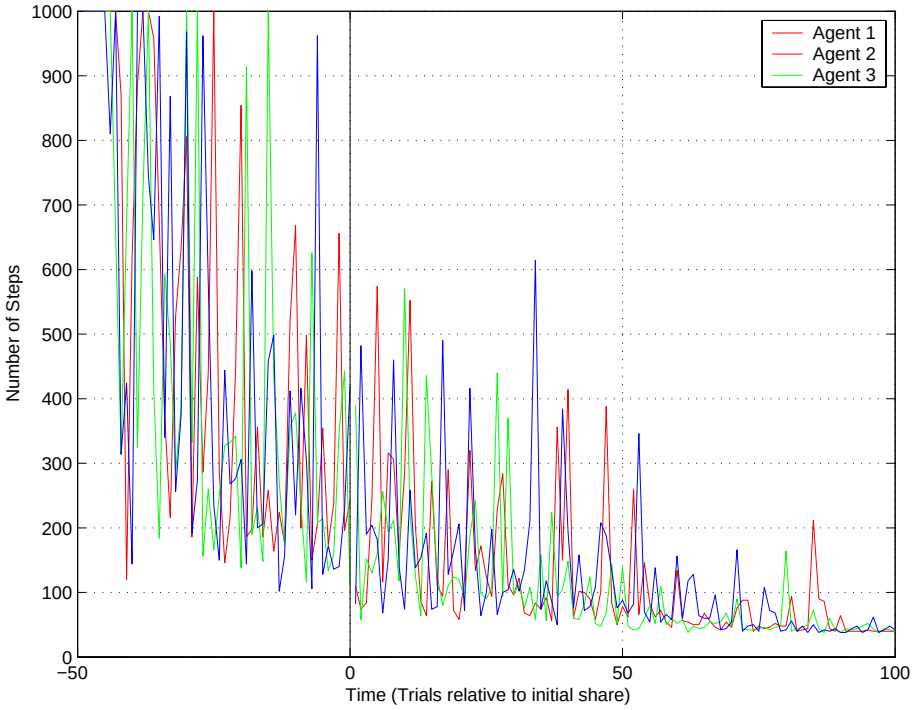


**Figure 28: Simulation 4b (general dynamic maze, p=0.1) performance plot**

These results show that in general, the Q-learning process was not severely affected by the dynamic nature of the world until the problem becomes in feasible, and in particular cooperative Q-learning did not seem to cause a decrease in performance when used in the dynamic world.

The reason for this is that the Q-learning method is very good at incremental repair. During the learning process, an entire field of Q-values is updated. If an agent finds an obstacle where there previously was not one, it simply updates the relevant Q-field, bounces off the obstacle and continues from a different state where other Q-values exist. In fact, as obstacles are increased, Q-learning does not show a significant loss in performance until the problem becomes infeasible as shown in Figure 27; after trial 55 a route from the start to the goal no longer exists but only then does performance decrease sharply.

## 9.2 Simulation 5: Performance in a 'Doors' Scenario

It was shown in section 9.1 that a dynamic world, in general, had very little impact on the performance of both individual and cooperative Q-learning. It was postulated that a scenario could be constructed in which the dynamic nature of the world *would* be detrimental to the performance of cooperative Q-learning. Such a scenario is presented here:

The maze shown in Figure 3 requires the agent to pass through one of three 'doorways' to reach the goal. Three agents are used, all of which carry out 200 individual trials. However the agents' individual learning is staggered, so that agent 1 learns from *t=-600* to *t=-400* while agent 2 learns from *t=-400* to *t=-200* and agent 3 learns from *t=-200* to *t=0*. During each of these three distinct periods, different doors are opened and closed. For the period where agent 1 is learning,

all three doors are open. For the period when agent 2 is learning, door 1 is closed, while for the period when agent 3 is learning, doors 1 and 2 are both closed. *For the test phase, only door 3 is open.*

This means that while agent one learns an optimal path through door 1 to the goal, this path is no longer feasible during the testing phase. Similarly, agent two learns a path through door 2, but this path is not open during the testing phase. Only the path learnt by agent 3 still exists during testing.

It was postulated that when cooperation occurs, each of the agents will be considered approximately equally expert in most states, and hence the converged but invalid Q values from agents 1 and 2 will adversely affect the performance of agent 3 whose Q-values are not only converged but also valid for the current configuration of the world.

The results for simulation 5 are shown from Figure 29 to Figure 31.

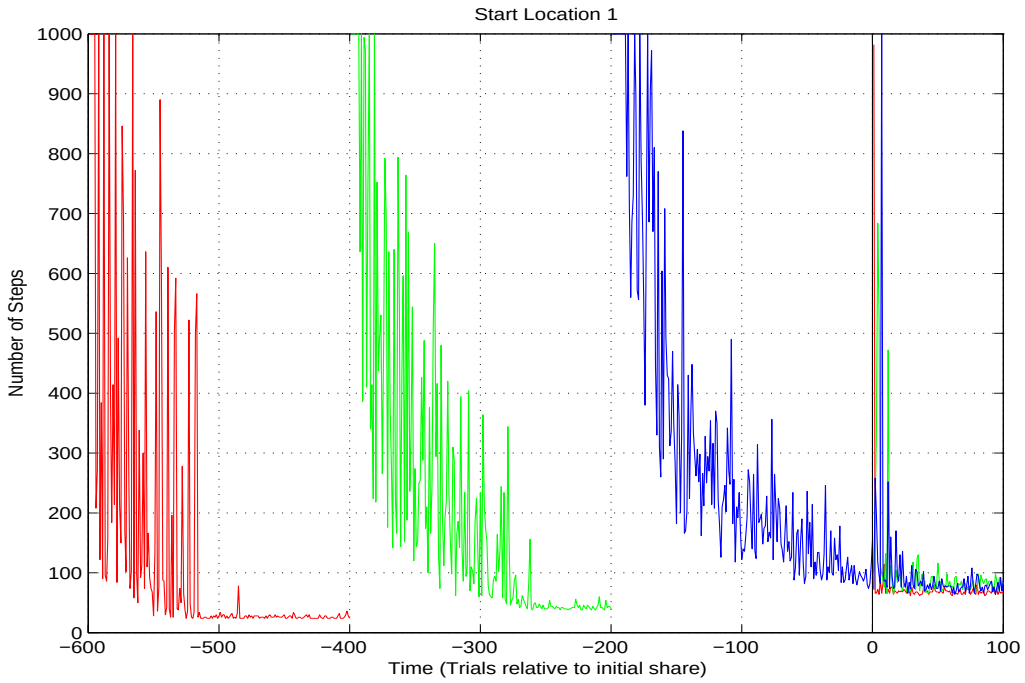


**Figure 29: Simulation 5 (doors scenario) performance plot**

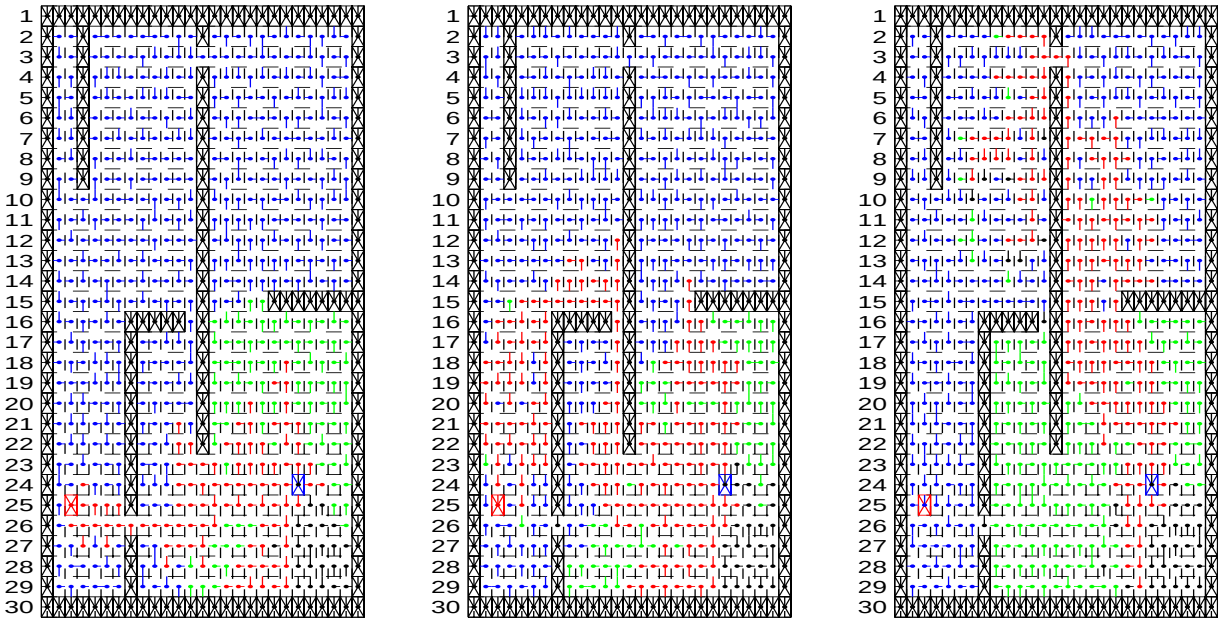

**Figure 30: Simulation 5 Q field after individual phase**

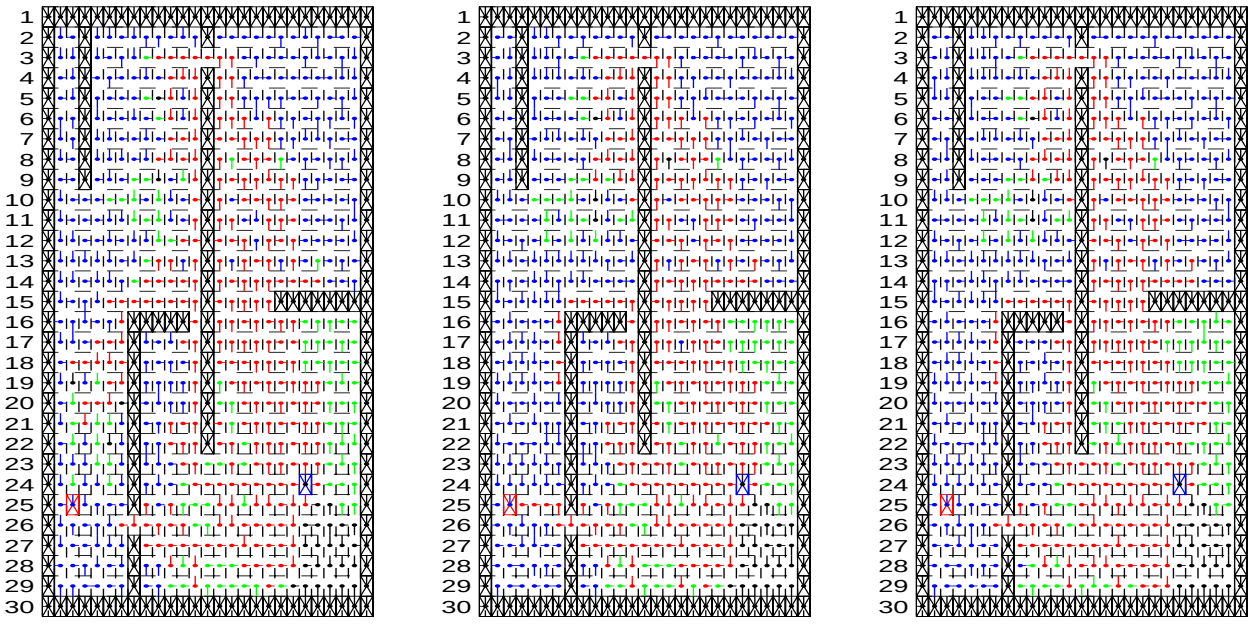

**Figure 31: Simulation 5 Q-field after cooperation**

Figure 29 shows that after an initial decrease in performance, all agents very quickly converged to the optimal path through door 3. This is a noteworthy result, since agents 1 and 2 have essentially invalid Q-fields and even though the weighted strategy sharing algorithm has no way of distinguishing between the valid and the invalid converged Q-fields, all three agents have almost optimal policies only 10 trials after cooperation.

This shows that cooperative Q-learning as described in section 3.3 is not significantly affected by a dynamic environment of this form. The reason for this is that, as described in section 9.1, Q-

learning alone is very good at incremental repair. Inspection of Figure 31 shows that agent 3 may be taken on a path which tries to go through door 2 due to the Q-values obtained from agent 2. However on discovering the door, the Q-values surrounding the door will be updated. After a few trials, the information about the door will have propagated to the surrounding cells, at which point an alternative action leading the agent to the correct path through door 3 will be selected by the agent.

In conclusion, a dynamic world even in this artificially constructed case causes only a temporary glitch in performance after cooperation and is not a serious impediment to the efficacy of cooperative Q-learning.

## 9.3 Simulation 6: Performance in a Contrived Scenario

In the previous scenario it was shown that an agent is able to 'repair' its Q-table after discovering the new obstacle where an open doorway had once been. The repair propagates back from the door to the start along the path of the agent. In the previous scenario, only a small amount of back-propagation had to occur before the agent found an alternative route to the goal.

In simulation 6, an extremely contrived scenario is created where the agent has to backpropagate almost the entire length of the path from the goal back to the start before an alternative route can be found. The map for this world can be seen in Figure 4.

In this simulation there are two agents which carry out 100 individual trials during staggered periods. During the first period, when agent 1 is learning, the top door is open and the bottom door is closed. During period when agent 2 is learning, the top door is closed and the bottom door is open. *During the test phase, the top door is closed and the bottom door is open.* Hence only agent 2 learns the path to the goal which is valid during the test phase.

The results for simulation 6 are shown from Figure 32 to Figure 34.

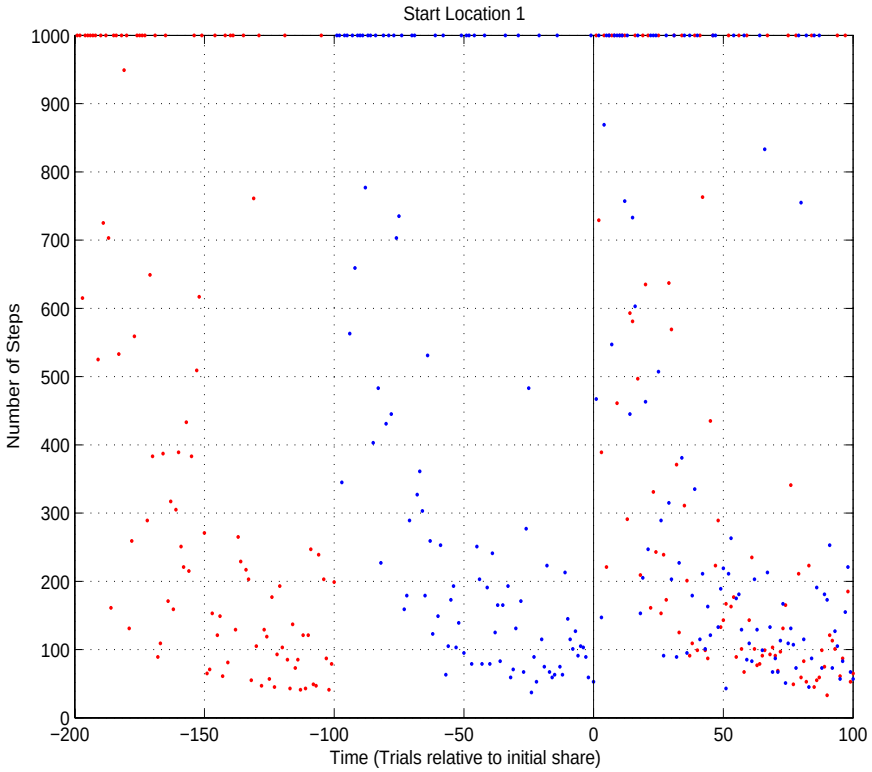


**Figure 32: Simulation 6 (contrived case) performance plot**

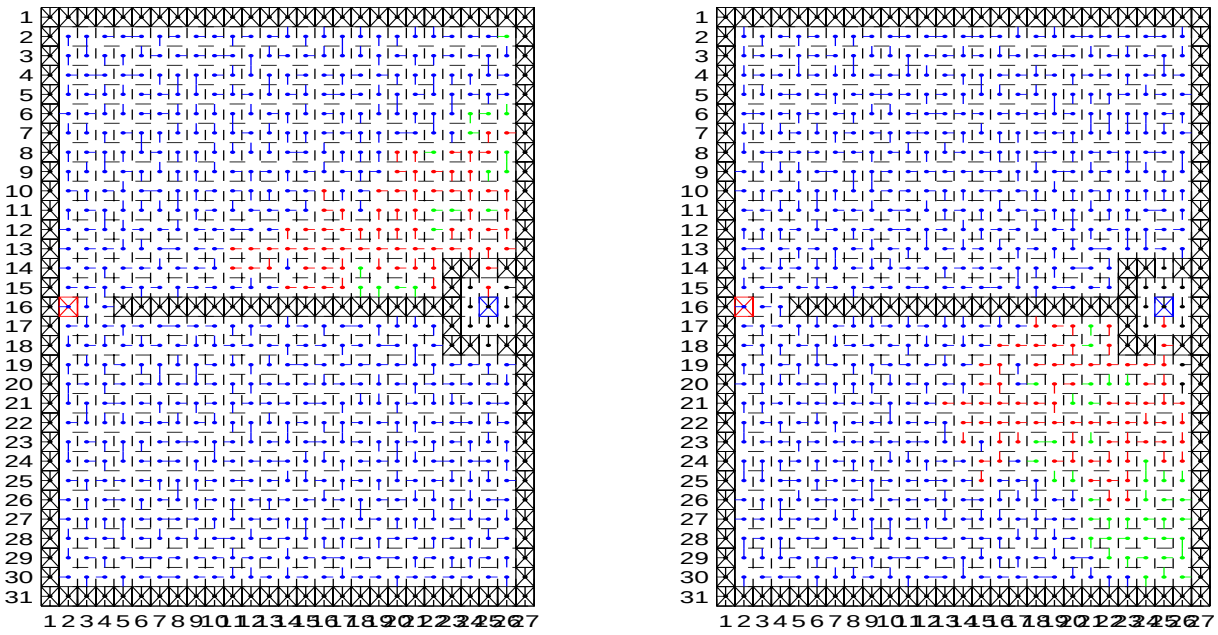

**Figure 33: Simulation 6 (contrived case) Q-field after individual phase**

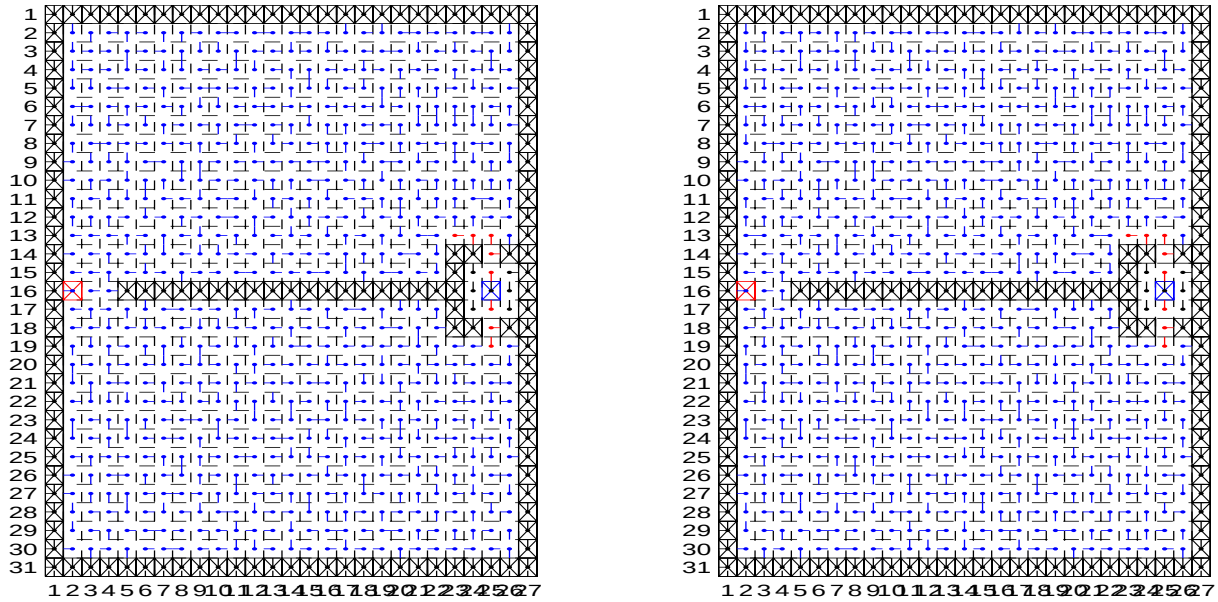

**Figure 34: Simulation 6 (contrived case) Q-field after cooperation**

Figure 32 shows that the performance of agent 2 is made much worse than the individual learning case after the Q-tables are shared, while the performance of agent 1 is as poor as would be expected given the change in the world. Note that in this case, the line plot has been replaced with points in order to highlight the fact that even 70 trials after cooperation, many trials for both agents hit the simulation limit of 1000 steps without reaching the goal.

The Q field in Figure 33 shows that each of the agents has learnt converged Q values corresponding exclusively to the side of the central barrier on which the door is open, as expected. Although after the top door is closed agent 1's Q values do not approximate the optimal Q* values, the weighted strategy sharing presented in 3.3 does not distinguish between the two agents who are approximately equally experienced. In fact, according to the expertness measure defined in 3.2, the agents will be considered most expert in states which they have traversed most often; these states may in fact be those where the agent has spent thousands of cumulative steps 'lost' rather than those where the agent has found a path to the goal. These unconverged but often traveresed states show up as blue pointers in Figure 33. Hence when the Q tables are shared, all of the converged Q values represented by the red pointers are lost, as shown in Figure 34. The performance of both agents is therefore very poor after cooperation as observed.

In conclusion, a contrived case was found where cooperation using the algorithm described in 3.3 caused the performance of both agents in the world to decrease significantly due to the dynamic nature of the world.

## *10. Discounted Expertness*

The failure of the cooperative Q-learning algorithm found in the previous section was due to the fact that both agents were assigned expertness values based entirely on the rewards the agent had received in a particular state. However only the learning which agent 2 had carried out was valid for the state of the world during the test phase.

It seems intuitive that in general, expertise gained by recent experiences is more valuable than that gained by experiences a long time in the past. In other words, expertness becomes less valuable over time.

An extension to the algorithm described in 3.3 is therefore proposed, in which the expertness value assigned to an agent for a particular state is discounted at every time step. This method is referred to in this report as *expertness discounting*. Expertness is now calculated in a recursive manner as shown below.

$$e_i\big|_t = \lambda e_i\big|_{t-1} + |R|_t$$

The discount factor λ is between zero and one and determines how quickly expertness due to past experiences loses its value.

The discounted expertness algorithm was implemented and used for the following simulations.

### 10.1 Simulation 7: Doors Scenario with Discounted Expertness

In simulation 7, the 'doors' scenario in simulation 5 was repeated except with the discounted expertness algorithm implemented. The results are shown in Figure 35 and Figure 36.

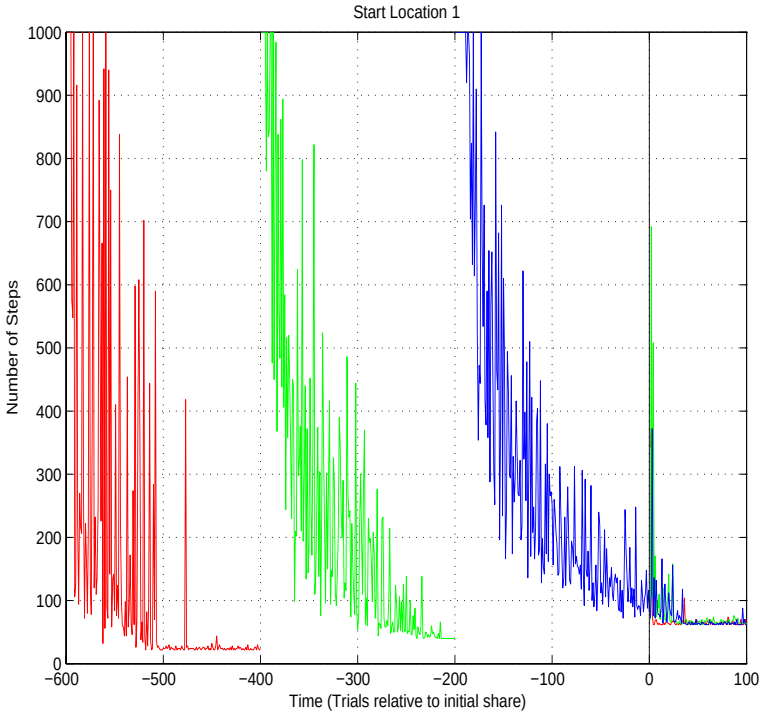


**Figure 35: Simulation 7: (Doors scenario with Discounted Expertness) performance plot**

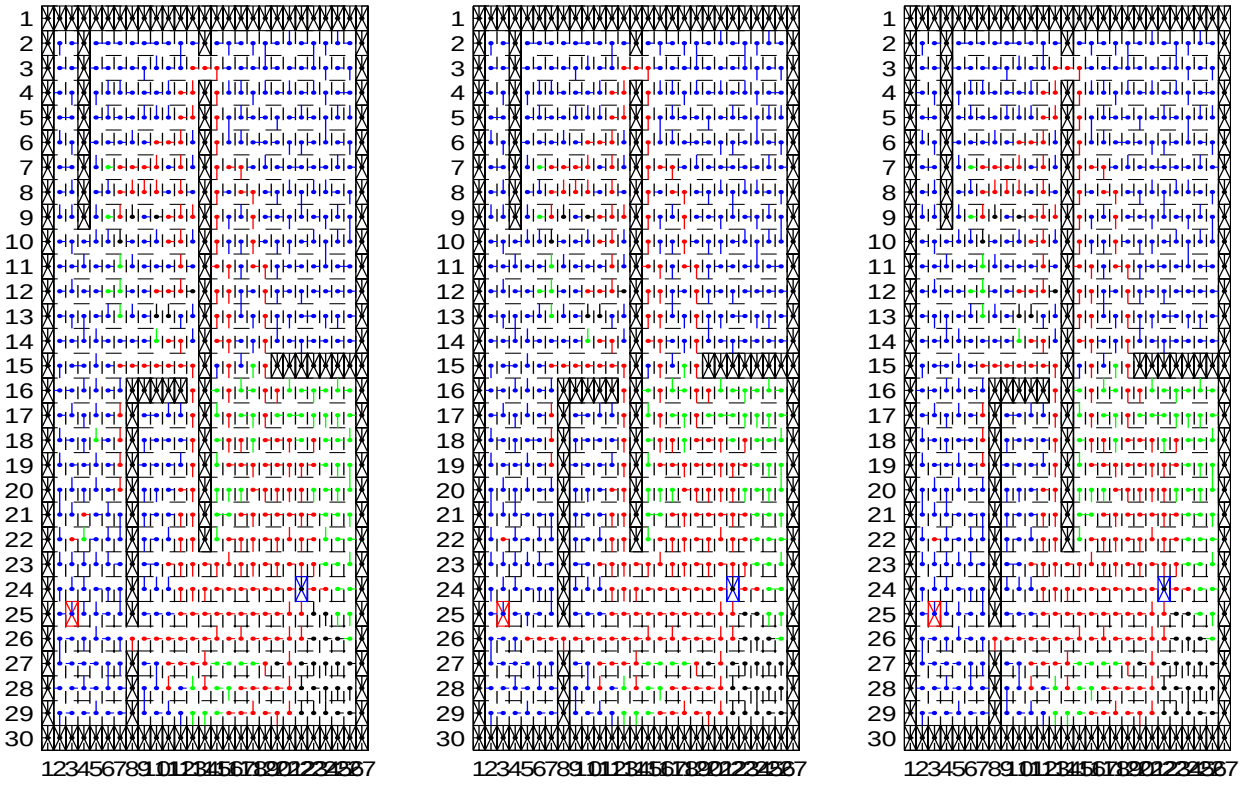

**Figure 36: Simulation 7: (Doors scenario with Discounted Expertness) Q-field after sharing**

Figure 35 shows that the performance is largely similar to that obtained without discounted expertness in Figure 29. However direct comparison with Figure 29 shows that the 'glitch' in

performance caused by the dynamic nature of the world directly after sharing Q values *has* been removed to some extent. Furthermore, note that the glitch in performance is now almost entirely in agents 1 and 2; these agents' Q tables before sharing were almost entirely false and hence such as drop in performance when exposed to the changed world is inevitable. *The performance of agent 3 is no longer adversely affected by cooperation with other agents.*

In conclusion, the use of discounted expertness has improved the performance of the cooperative Q-learning algorithm in a specially-constructed dynamic world by reducing the negative effects of the changing nature of the world. However since these negative effects are very brief, the overall impact of discounted expertness is limited.

## 10.2 Simulation 8: Contrived Scenario with Discounted Expertness

In simulation 8, the discounted expertness method was tested with the contrived scenario used in simulation 6. It was postulated that since the dynamic nature of the world severely reduces the performance of the cooperation algorithm in this case, there is a greater opportunity for improvement using the discounted expertness method.

Results for the simulation are shown from Figure 37 to Figure 39.

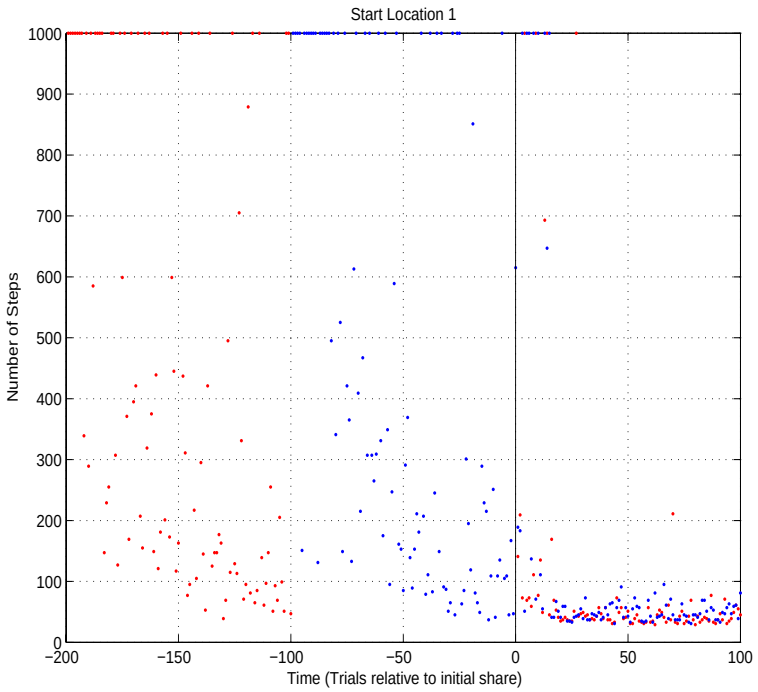


**Figure 37: Simulation 8 (contrived case with discounted expertness) performance plot**

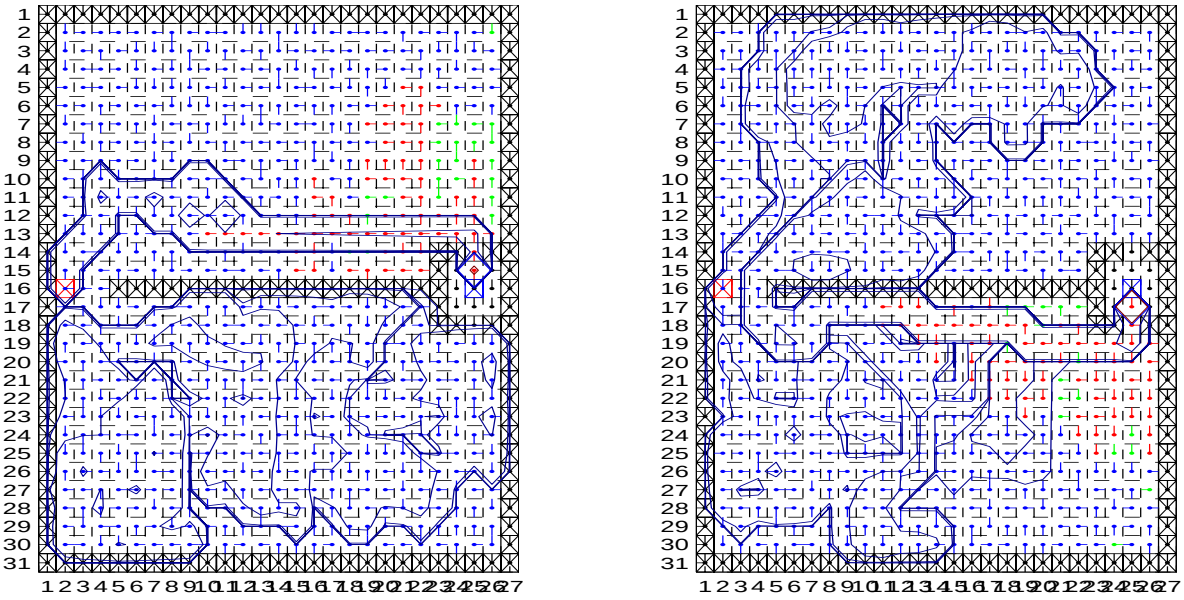

**Figure 38: Simulation 8 (contrived case with discounted expertness) after individual trials**

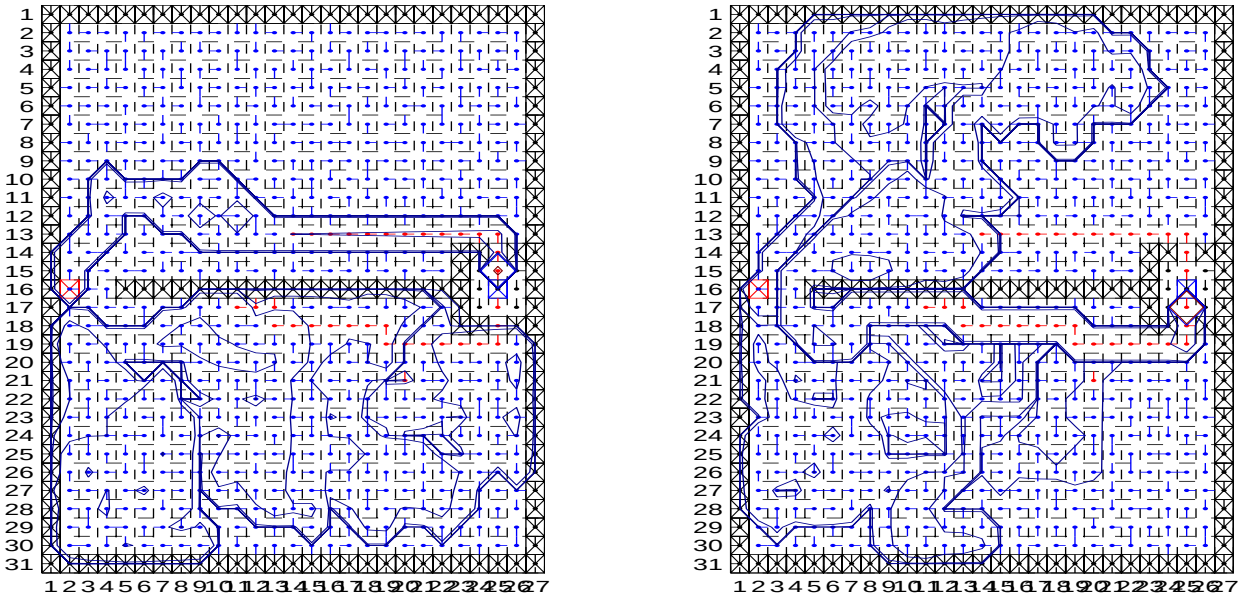

**Figure 39: Simulation 8 (contrived case with discounted expertness) Q field after sharing**

Comparing Figure 32 with Figure 37 shows that the performance of both agents has been improved enormously by using the discounted expertness method. Whereas before, several trials reached the limit of 1000 steps without reaching the goal even 70 trials after cooperation, now, with discounted expertness, both agents have converged to nearly optimal policies after 20 trials.

Figure 39 shows the Q fields for both agents after sharing. The lines shown are *contours of expertness* drawn for each agent. These contours are intended to highlight the regions in which an agent is expert. Comparing Figure 39 to Figure 38 shows that instead of removing all the

converged information about the optimal path to the goal, the cooperation stage has retained some of this information.

There are two interesting features to note, however. Firstly, some converged Q values from the upper half of the map have been retained even though these are no longer valid. Inspection of the expertness contours shows that agent 2 has almost zero expertness in the upper right quadrant of the map where these false Q values have been retained. Even though agent 1's expertness in this region is heavily discounted, it is still greater than that of agent 2. Hence some of agent 1's invalid Q values are retained.

Secondly, many of the correct Q values from agent 2 are *not* retained after the share. This is because, as the expertness contours show, agent 2 has a low expertness in certain areas where the Q field is converged. This is because once a near-optimal path is found, this path will be repeated, increasing the expertness along the nominal path but allowing the expertness in other regions to decay due to the expertness discounting. On the other hand, agent 1 has been 'lost' in the same region for many thousands of steps and has hence accumulated a great deal of expertness in that zone.

Part of this effect is due to the fact that the expertness measure used is based only on rewards. Hence in a goal-seeking problem, an agent can only become expert in a non-goal state by visiting that state many times. This does not take into account the added value of states where Q values are high because a route to the goal has been found.

In conclusion, the discounted expertness extension significantly improves performance in a particular dynamic world contrived in such a way that the dynamic nature of the world causes cooperative Q-learning to fail in the absence of discounted expertness.

## 11. Conclusion

The following are overall conclusions from the project:

Expertness based cooperative Q-learning with specialized agents as presented in reference 2 with state based specialisation gives an improvement in performance for two distinct cases. The first of these cases is when agents carry out individual learning in separate areas of the state space and then are expected to perform in areas with which they are unfamiliar. The second is when agents explore a similar region of the state space, but have significantly different experience levels.

In both of these cases, the result is somewhat obvious; that the agents which have little or no knowledge about the particular part of the state space effectively acquire the Q values of the experienced agent. As expected this improves the performance of the inexperienced agent in the relevant areas of the state space, but only up to the level of the most expert agent.

In the two cases where the algorithm is strong, the large difference in expertness values between agents means that a simpler weighting method, Most Expert, is likely to be just as effective as the Learning from Experts method. It was shown that this was the case for a representative simulation.

It should be noted that the results shown in this report are a representative subset of the simulations carried out when justifying the conclusions made. Although the nature of the algorithm means that results are problem specific, enough additional tests not displayed here were carried out to strongly suggest that the conclusions made are valid at least for the goal-seeking problem in general.

It was shown that Q-learning in general, and cooperative Q-learning in particular, is largely unaffected by dynamic environments except in specially constructed cases. In these contrived cases, cooperation actually reduced the performance of all agents compared to individual learning.

In response to this, the *discounted expertness* concept was introduced. This was shown to give significant improvements in performance in the aforementioned contrived cases. Although this seems like a weak statement, the following is important to note: In order for a cooperation algorithm to be of use, it must at worst not decrease the performance of the agents compared to individual learning without cooperation. If the algorithm decreases performance in certain pathological cases, as in the case of the expertness based cooperation algorithm, then its application must always be qualified by a supervisory check for these pathological cases. By using the discounted expertness extension, at least some, if not all, of these problematic cases are avoided.

On a separate note, during the course of the project, an issue relating to the use of the expertness measures in maze worlds was identified; namely that away from the goal state, expertness in a state can only be accrued by visiting the state. When sharing, no preference is given to Q values corresponding to propagation of information from the goal; these values are obviously of much

greater value than those where an agent has been lost but traversed the same state many times. While Ahmadabadi et al. do consider the entropy of the Q table as a measure of expertness, without much success in improving performance, perhaps in the maze case at least measuring expertness using the *magnitude* of the Q value in each state would be an interesting avenue to pursue.

The conclusions above have only been shown to hold for the specific cases for which simulations were carried out. Intuitively, however, aspects such as the cases where the existing algorithm will give a performance improvement or not, would apply to many Q-learning scenarios. Investigating how broad reaching the conclusions are would be another interesting area of research.

## *12. References*


**1:** *Expertness Based Cooperative Q-Learning* Ahmadabadi, M. N. and Asadpour, M. IEEE Transactions on Systems, Man and Cybernetics – Part B: Cybernetics, Vol. 32, No. 1 February 2002

**2:** *An Extension of Weighted Strategy Sharing in Cooperative Q-Learning for Specialized Agents* Eshgh, A. M. and Ahmadabadi, M. N. Proceedings of the 9th International Conference on Neural Information Processing (ICONIP'02), Vol. 1. 2002.

**3:** *Multi-Agent Reinforcement Learning: Independent vs. Cooperative Agents* Tan, M. 1993.

**4:** *A Complexity Analysis of Cooperative Mechanisms in Reinforcement Learning.* Whitehead, S. D. Proceedings of AAAI-91. 1991.