School of Informatics, University of Edinburgh

Institute of Perception, Action and Behaviour

Could Active Perception Aid Navigation of Partially Observable Grid Worlds?

by

Paul Crook, Gillian Hayes

Informatics Research Report EDI-INF-RR-0180

School of Informatics **September 2003**
http://www.informatics.ed.ac.uk/

Could Active Perception Aid Navigation of Partially Observable Grid Worlds?

Paul Crook, Gillian Hayes

Informatics Research Report EDI-INF-RR-0180

SCHOOL *of* INFORMATICS
Institute of Perception, Action and Behaviour

September 2003

appears in Proceedings of ECML 2003

Abstract :

Due to the unavoidable fact that a robot's sensors will be limited in some manner, it is entirely possible that it can find itself unable to distinguish between differing states of the world (the world is in effect partially observable). If reinforcement learning is used to train the robot, then this confounding of states can have a serious effect on its ability to learn optimal and stable policies. Good results have been achieved by enhancing reinforcement learning algorithms through the addition of memory or the use of internal models. In our work we take a different approach and consider whether active perception could be used instead. We test this using omniscient oracles, who play the role of a robot's active perceptual system, in a simple grid world navigation problem. Our results indicate that simple reinforcement learning algorithms can learn when to consult these oracles, and as a result learn optimal policies.

Keywords : Active Perception, Perceptual Aliasing, Partially Observability, POMDP, Reinforcement Learning, Grid World Navigation

Copyright © 2003 by Springer-Verlag; publication rights retained by University of Edinburgh for academic and research purposes

The authors and the University of Edinburgh retain the right to reproduce and publish this paper for non-commercial purposes.

Permission is granted for this report to be reproduced by others for non-commercial purposes as long as this copyright notice is reprinted in full in any reproduction. Applications to make other use of the material should be addressed in the first instance to Copyright Permissions, School of Informatics, The University of Edinburgh, 2 Buccleuch Place, Edinburgh EH8 9LW, Scotland.

Could Active Perception Aid Navigation of Partially Observable Grid Worlds?

Paul A. Crook and Gillian Hayes

Institute of Perception, Action and Behaviour,
School of Informatics, University of Edinburgh,
5 Forrest Hill, Edinburgh. EH1 2QL, UK
{paulc, gmh}@dai.ed.ac.uk
http://www.dai.ed.ac.uk/homes/paulc/index.html

Abstract. Due to the unavoidable fact that a robot's sensors will be limited in some manner, it is entirely possible that it can find itself unable to distinguish between differing states of the world (the world is in effect partially observable). If reinforcement learning is used to train the robot, then this confounding of states can have a serious effect on its ability to learn optimal and stable policies. Good results have been achieved by enhancing reinforcement learning algorithms through the addition of memory or the use of internal models. In our work we take a different approach and consider whether active perception could be used instead. We test this using omniscient oracles, who play the role of a robot's active perceptual system, in a simple grid world navigation problem. Our results indicate that simple reinforcement learning algorithms can learn when to consult these oracles, and as a result learn optimal policies.

1 Introduction

Partially observable environments cause particular problems when reinforcement learning is used to learn the task. Reinforcement learning algorithms associate rewards and actions with states, but in partially observable environments the true state can be masked. This makes it virtually impossible (apart from some very simple worlds [1]) for basic 1-step reinforcement learning algorithms to converge to optimal policies [2, pp73–78]. Work in this area has shown that when learning algorithms are augmented with memory [3] or the ability to learn a model of their world [4,5], they can find optimal solutions to this type of problem. Our aim is to examine an alternative approach. Rather than equipping the learning algorithms with memory or the ability to model their environment, we propose to equip agents or robots with sensors which they can actively control, and hope to demonstrate that the learning algorithms can learn to use this active perception to find optimal solutions.

This paper considers the questions: (i) could an agent learn to use an active perceptual system?, (ii) is there any benefit for an agent in using such a system to disambiguate its current state? To study the fundamentals of the problem we consider simulated agents moving around deterministic grid worlds, for example

Proceedings of the 14^{th} European Conference on Machine Learning (ECML-2003)
©Springer-Verlag http://www.springer.de/comp/lncs/index.html

Sutton's Grid World (Fig. 1). In order to focus on the questions raised above we make the assumption that it is possible to give an agent an active perception system which it could use to disambiguate the current state. To this end we introduce oracles which the agent can consult. These play the role of active perception systems which can disambiguate the agent's current state perfectly, see section 2.4. Thus the hypotheses that we attempt to test are:

(i) Reinforcement learning algorithms can learn when to use additional resources in order to determine an agent's true state in a partially observable grid world.
(ii) Resolving the agent's true state in a partially observable world allows reinforcement learning algorithms to learn optimal policies.

2 Background

2.1 Reinforcement Learning & Partially Observable Worlds

Whitehead [2, p72] identifies two distinct problems which occur when using reinforcement learning with robotic systems in partially observable environments: local and global impairment.

As an example of local impairment consider a robot standing at one of two similar looking T-junctions. It is unable to distinguish between the two junctions and hence regards them as the same state. We use the phrase *aliased state* to refer to such states which appear to be identical but in the underlying model are distinct. The robot's policy when learnt using reinforcement learning can only link a single action to the perceived single state, thus it will execute the same action at both of the T-junctions. If at one T-junction the optimal action is to turn left and at the other it is to turn right then the single action learnt by the policy will be wrong in one of the two cases[1]. More generally, when learning a state-action policy where there exist aliased states, the policy will sometimes select actions which are inconsistent with the actual underlying state.

Global impairment occurs where inconsistent state values produced by aliased states are used to update the state values of otherwise consistent states. This occurs independently of whether the action selected in the aliased states is inconsistent. As an example, consider again the robot standing at one of the two T-junctions. If the optimal action at both T-junctions is the same, *e.g.* turn left, then action selection is not a problem as the policy can link this single action with the single perceived state, however there is still a problem with representing the value of the aliased states. In reinforcement learning the policy is generally represented by storing a value for each state or each state-action pair. These values indicate the utility of being in that state (or selecting a given action from that state). Given that the two T-junctions are perceived as one and the same location, a single state value or single set of state-action values will be stored. The stored value(s) will be updated to represent the value of being at both of the T-junctions and hence over time become a weighted[2] average of the true value

[1] Assuming the only possible actions are turn left or right.
[2] Weighted by the number of times each state is visited.

©Springer-Verlag http://www.springer.de/comp/lncs/index.html

of each of the underlying states. Now, if one junction is far from the goal and the other is very close to the goal, their averaged value will make the junction that is far from the goal appear more attractive than it should, while the junction near the goal will appear to be a less valuable state than it should. If 1-step backup of state values is employed, such as in SARSA or Q-learning [6, p146, p149], then these averaged state values will be used to update those states around the T-junctions causing errors in their state values and possibly also in the selection of actions. These errors in the state values can propagate from state to state affecting the whole of the robot's policy.

2.2 Satisficing & Optimal Memoryless Policies

Littman [7] considered learning state-action policies in partially observable environments and introduced the useful concepts of *satisficing* and *optimal memoryless policies*. A policy is said to be satisficing if independent of its initial state an agent following this policy is guaranteed to reach the goal. A memoryless policy is a policy which selects an action based solely on the current sensation. SARSA and Q-learning are examples of reinforcement learning algorithms which work on exactly this basis. If the performance of a policy is measured by the number of steps taken to reach the goal summed over all possible initial states of the agent (*total steps*), then an optimal policy is one that achieves the minimum total steps to the goal and an optimal memoryless policy is a policy that achieves the minimum number of total steps which can be achieved by a memoryless policy.

Singh et al.[8] proved that an optimal memoryless policy is arbitrarily worse than the optimal policy which can be achieved in the absence of aliased states. Despite this result, Littman [7] used branch and bound techniques to demonstrate that it is possible to find optimal memoryless solutions for various grid world navigation problems, and for the range of grid worlds examined, the optimal memoryless policies did not take an unreasonable number of extra steps.

2.3 Eligibility Traces

Whitehead [2] indicates that global impairment prevents 1-step backup reinforcement learning algorithms from being able to learn stable and optimal policies for partially observable environments. However Loch and Singh [9] showed that the simple addition of eligibility traces to reinforcement learning allows it to find optimal memoryless policies in grid world navigation problems. Eligibility traces appear to avoid the problem posed by global impairment by updating a chain of previously visited states.

2.4 Active Perception

By active perception we mean a perceptual system which an agent can direct in order to vary the input it receives from the world. An obvious example is a video camera mounted on a robotic arm. An agent equipped with such a

©Springer-Verlag http://www.springer.de/comp/lncs/index.html

camera can direct the movement of the arm and thus obtain different views of its surroundings. In the context of grid worlds, active perception would involve the agent being able to choose which grid squares make up its input state. At this stage however we are interested in simplifying the problem in order to understand the underlying dynamics. Therefore rather than equip the grid world agents with active perception systems which they would have to learn to coordinate, we have instead provided them with oracles.

Introducing oracles which the agents can consult is a useful test. If an agent can learn when to make appropriate use of an oracle, then it should be able to learn when to use an active perceptual system. On the other hand, if it fails to make use of an oracle, it seems unlikely that it could learn to use an active perception system, especially as the latter may require the coordination of many more actions. In the experiments presented below we have used two types of oracle. These correspond to two possible questions "where am I?" or "where should I go?". The first, the *State Oracle*, can on request provide the agent with an unambiguous state representation of its current location. The second, the *Action Oracle*, explicitly directs the agent based on a known optimal solution (the solution used is that indicated by the arrows in Fig. 1).

Our hypothesis, that agents using active perception (or in this case oracles) together with eligibility traces should be able to learn optimal solutions, stems from the observation that oracles resolve the problem of local action selection, and eligibility traces address the issue of global impairment.

We do not currently envisage using oracles when implementing this technique on real robots. They are introduced as a useful simplification to aid understanding of the underlying problems. However, it is possible to imagine scenarios where resource constraints might make the use of oracles worthwhile, *e.g.* mass produced military robots with limited computational power and sensors, which have access to a central computer to aid their exploration. To avoid overloading the central computer or to minimise the number of transmissions they make, these robots might only call on the central computer (their oracle) when unable to determine their location independently.

3 Experiments

The experiments presented here use Sutton's Grid World (Fig. 1) which consists of a 9×6 grid containing various obstacles and a goal in the top right hand corner (indicated by an asterisk). An agent in this world can choose between four *physical actions*; move north, south, east and west. The agent receives a reward of -1 for each action which does not move it directly to the goal state and a reward of 0 for moving directly to the goal state. State transitions are deterministic and each action moves it one square in the appropriate direction. If an agent selects a physical action whose execution is obstructed by an obstacle or wall, then the agent's location remains unaltered but it receives the same -1 reward as above. When the agent reaches the goal state it is relocated to a uniformly random start state.

©Springer-Verlag http://www.springer.de/comp/lncs/index.html

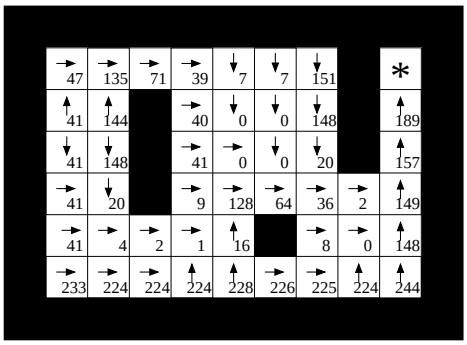

Fig. 1. Sutton's grid world. Values indicate the observations obtained by an agent observing the eight squares surrounding its current location (Eight Adjacent Squares Agent). Arrows shows an example optimal policy

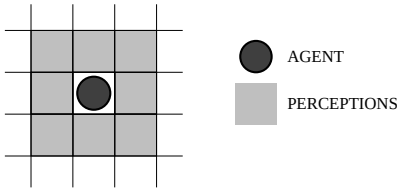

Fig. 2. Eight Adjacent Squares Agent has a state representation formed by observing the eight squares surrounding its current location

Littman [7] modified Sutton's original problem by introducing an agent whose state representation is formed by observing the eight squares adjacent to its current location (Fig. 2), an *Eight Adjacent Squares Agent*. This is opposed to an agent whose state representation is its location in the world given in Cartesian coordinates and is more representative of the problem faced by a robot in a building. For an Eight Adjacent Squares Agent there are multiple locations that are aliased; Fig. 1 shows indicative values which represent the perception of such an agent. Some of the aliased states cause the agent to learn inconsistent actions, *e.g.* perception 148 occurs in three locations: (i) four squares directly below the goal, (ii) near the middle of the obstacle which is to the left of the goal and (iii) near the middle of the far obstacle. As can be seen from Fig. 1 the optimal solution (indicated by the arrows) requires a different action in one of the three occurrences of this perceptual state. Other aliased states cause only global impairment, *e.g.* perception 2 which occurs just below the obstacle near the goal and just below the far obstacle. In this case the optimal action is always the same, *i.e.* move east, but the aliased state values cause problems as one location is very near the goal and the other is quite a distance away.

©Springer-Verlag http://www.springer.de/comp/lncs/index.html

We used three types of agent:

(i) *Eight Adjacent Squares Agent.* An agent whose state representation is formed by observing the 8 squares adjacent to its current location, as shown in Fig. 2.

(ii) *Action Oracle Agent.* An agent who has the same state representation as the Eight Adjacent Squares Agent but has an additional action allowing it to ask the Action Oracle in which direction it should go. The agent then immediately executes the action specified by the oracle.

(iii) *State Oracle Agent.* An agent whose normal state representation is the same as the Eight Adjacent Squares Agent, but who can ask the State Oracle where it is. On asking the oracle the agent receives a state representation that corresponds to its absolute location in the world given in Cartesian coordinates.

The latter two agents receive a reward of -1 for selecting the action to ask their respective oracles a question. This reward is in addition to the reward for executing any subsequent actions. The way this works in practice is that the Action Oracle Agent, on asking its oracle where to go, transitions to an adjacent location in the grid world based on the optimal action from its current location, and receives a reward of -2 (-1 for asking the oracle and -1 for the action executed). The State Oracle Agent, on asking its oracle where it is, sees a state transition from its normal representation of its current location to an absolute coordinate system representation of the same location, receiving a penalty of -1 for asking the oracle. It then has to learn the optimal action to execute from this new state representation. Once it selects a physical action, its state representation reverts back to normal, *i.e.* it is formed by observing the 8 squares adjacent to its new location. This dual representation of the State Oracle Agent's location in the world more than doubles the state space which it needs to explore.

A range of reinforcement learning algorithms were used: SARSA, Q-learning, SARSA(λ) with replacement traces, Watkins's Q(λ) with accumulating traces. For details of the learning algorithms see [6, p146, p149, p181, p184] respectively. All of the learning algorithms continuously updated their policies. Actions are selected greedily using the current policy with a probability of $(1 - \epsilon)$. In cases where actions have the same value, ties were broken at random. In the remaining ϵ cases the action executed was selected randomly between all the available actions. In all cases above, random selection is uniform across all possibilities. State-action values for all the learning algorithms were initiated at zero.

The following values were used for all learning algorithms: (i) learning rates (α) of 0.1 and 0.01, (ii) discount rate $\gamma = 0.9$, (iii) probability of random action (ϵ) started at 20% and decayed linearly reaching zero at the $200,000^{th}$ action-learning step, thereafter it remained at zero. For the learning algorithms SARSA(λ) and Watkins's Q(λ) a range of values were tried for the eligibility trace decay rate (λ); from 0.01 to 0.9. Results are shown for $\lambda = 0.9$ which was in general found to perform the best.

©Springer-Verlag http://www.springer.de/comp/lncs/index.html

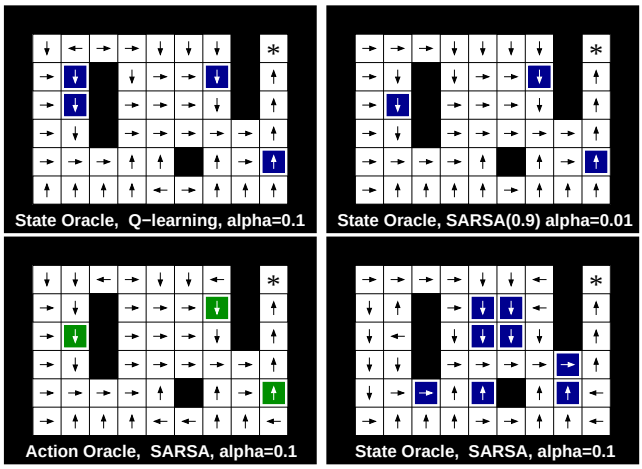

Fig. 3. Example policies learnt after 400,000 action-learning steps. Arrows indicate the physical action selected by the policy. Squares containing filled blocks indicate where the policy is to consult the oracle in order to select the shown physical action

4 Evaluation

To test the first hypothesis we ran the agents for 400,000 action-learning steps and looked at example policies learnt by individual agents to examine if they were making appropriate use of the oracles.

To evaluate the second hypothesis we again ran the agents for 400,000 action-learning steps but evaluated the current policy after every 1000 action-learning steps. The policy is evaluated by executing it greedily and determining the total steps required to reach the goal from every possible starting position. Separate counts were kept of physical actions and requests to the oracle. The agent is limited to a maximum of 1000 steps (total of both physical actions and requests to oracles) to reach the goal from each starting position. There are 46 non-goal starting states in Sutton's Grid World, so a policy that fails to reach the goal from all of them would have a maximum total number of steps of $46,000$. Each combination of agent, world, learning algorithm and values of α and λ was repeated 100 times giving 100 samples per data point.

5 Results

Fig. 3 shows examples of the policies learnt for both Action and State Oracle Agents, using SARSA, Q-learning and SARSA(λ). Grid squares containing filled blocks indicate where the policy is to consult the oracle. Of the examples shown three have learnt that they achieve a better solution if they consult their respective oracle when they are in the three squares labelled 148 in Fig. 1. As was discussed in section 3, the squares labelled 148 are perceived to be identical but the action required for an optimal solution is not the same in each location. In the fourth example in Fig. 3 the agent consults the State Oracle when in

©Springer-Verlag http://www.springer.de/comp/lncs/index.html

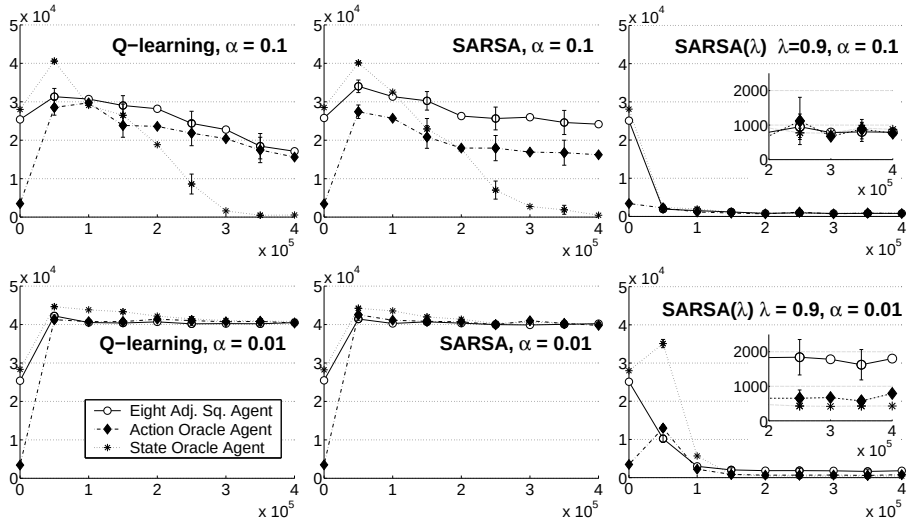

Fig. 4. Plot of mean total steps (sum of physical actions and requests to oracles) found when policies were evaluated, versus action-learning steps. To simplify plots data points are only shown for every 50,000 action-learning steps. Bars indicate 95% confidence intervals. The two inserts show enlargements of the tail of their respective plots

the squares labelled 0, 2 and 16 in Fig. 1. The grid squares labelled 0 require different actions to be executed in different locations. The squares labelled 2 are aliased but require the same action to be executed in both locations. Square 16 is not aliased. Of the examples shown, the top right (State Oracle, SARSA(λ) with $\lambda = 0.9$ and $\alpha = 0.01$) is an optimal solution in terms of physical actions.

Fig. 4 allows comparison of the mean total steps (sum of physical actions and requests to the oracles) of the three agents. Plots are shown for each of the learning algorithms with the exception of Watkins's Q(λ). Plots for Watkins's Q(λ), which are not shown due to space constraints, are very similar to those for SARSA(λ). The top three plots show results for $\alpha = 0.1$, the lower three plots show results for $\alpha = 0.01$.

For $\alpha = 0.01$ the two 1-step backup algorithms (SARSA and Q-learning) fail to learn reasonable policies using any of the agents. The mean total steps in these six cases remains just below the maximum of 46,000 steps. Results for SARSA and Q-learning are better when $\alpha = 0.1$, with the mean total steps for all three agents falling as learning progresses. Even so, the Eight Adjacent Squares Agent struggles to learn reasonable solutions. This is expected as 1-step backup of state values will cause global impairment of the learnt policies. The curve for the Action Oracle Agent is better in the case of SARSA but almost identical to the Eight Adjacent Squares Agent for Q-learning. The most significant difference is shown by the State Oracle Agent. It is slower to learn initially, due to the increase in the state space caused by the State Oracle (see section 3), but in the longer term it achieves much better results than either of

©Springer-Verlag http://www.springer.de/comp/lncs/index.html

the other two agents. In the two plots for SARSA(λ), the total mean steps of all three agents reduce rapidly, convergence occurring more quickly with $\alpha = 0.1$ than $\alpha = 0.01$, as would be expected. Inserts, which show the tail of both of these plots, indicate that for $\alpha = 0.1$ there is no distinction between the mean total steps for the three types of agent. However, for $\alpha = 0.01$ the average policy learnt by the eight adjacent squares agent is worse than that for the two agents which can use oracles.

To get a clearer picture of what is occurring we classify the policies that have been learnt into five categories: optimal, better than memoryless optimal, memoryless optimal, other satisficing and non-satisficing. We define these terms specifically for Sutton's grid world in terms of the total physical actions taken when the policy is evaluated. The optimal policy for Sutton's grid world is 404 physical actions. Littman [7] showed that the optimal memoryless solution for Sutton's grid world is 416 physical actions. Note that these terms are defined in terms of physical actions and exclude requests to oracles. This is in order to make a level comparison between agents with oracles and those without. Littman's [7] definition of a satisficing policy is one that reaches the goal from all possible start states. Our measure of satisficing is stricter than this requirement as the agent is limited to 1,000 actions from each start state, after which the run is truncated and the policy deemed to be non-satisficing. Note that this measure includes requests to oracles as it is possible for an agent to select no physical actions and spend all its time talking to the oracle. The other satisficing class contains those policies that are satisficing but exceed the number of physical action steps required by the other categories. The five categories are summarised in table 1.

Fig. 5 shows the categorisation of policies from 100 trials over the course of 400,000 action-learning steps. Top left shows the policies learnt by the State Oracle Agent using Q-learning with $\alpha = 0.1$. By the end of the trials the majority of policies are classified as other satisficing (72 policies), with 4 that are memoryless optimal, 20 that are better than memoryless optimal and 4 non-satisficing. No optimal policies have been learnt. These results are better than expected as the State Oracle only addresses the issue of local action selection, not that of global impairment. The success of this agent with 1-step backup learning algorithms is, however, heavily dependent on the value of α as no satisficing policies are learnt for $\alpha = 0.01$. The categorisation of policies learnt by the State Oracle Agent using SARSA (not shown) is very similar with final results of 79 other satisficing, 6 memoryless optimal, 14 better than memoryless optimal, and 1 non-satisficing.

Fig. 5 also shows the categorisation of policies for SARSA(λ) for each of the three agents. Separate plots are shown for $\alpha = 0.1$ and 0.01 for the Action Oracle and State Oracle Agents. For the Eight Adjacent Squares Agent, which has no access to an oracle, with $\alpha = 0.1$ the majority of solutions are memoryless optimal, 83 at the end of the 100 trials, with 4 other satisficing policies and 13 non-satisficing. The results are similar for this agent with $\alpha = 0.01$ (not shown), 68 memoryless optimal, 11 other satisficing and 21 non-satisficing. This is in line with our expectations, as Loch and Singh [9] showed that the use of eligibility traces allowed agents to find optimal memoryless solutions.

©Springer-Verlag http://www.springer.de/comp/lncs/index.html

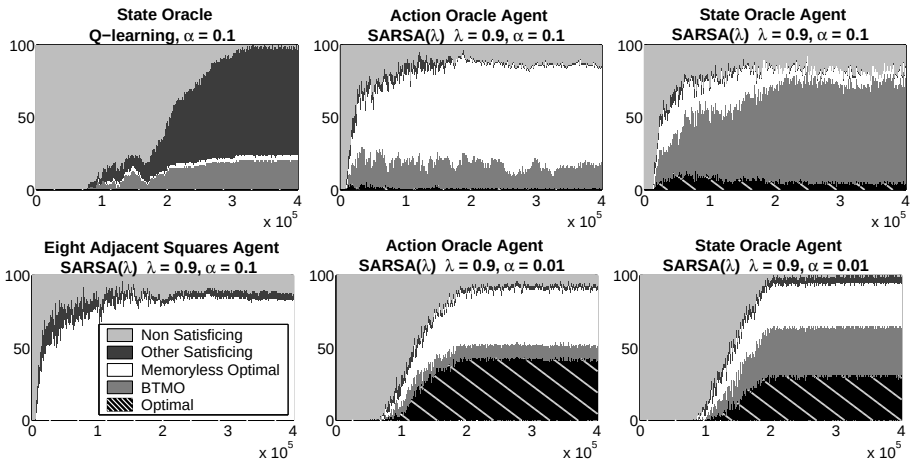

Fig. 5. Plots show categorisation of policies versus action-learning steps. Height of shaded areas indicate numbers of policies out of a total of one hundred that fall into each classification

Table 1. Policy categories for Sutton's Grid World

Goal Reached From All Start States	Total Physical Actions	Policy Category
yes	404	Optimal
yes	$405 - 415$	Better Than Memoryless Optimal (BTMO)
yes	416	Memoryless Optimal
yes	> 416	Other Satisficing
no	-	Non-Satisficing

With $\alpha = 0.1$ the Action Oracle Agent initially generates a small number of optimal policies, however these quickly disappear indicating that they are not stable and by the end the majority of policies (65) are memoryless optimal, 19 are better than memoryless optimal, 1 other satisficing, and 15 non-satisficing. The State Oracle Agent learns a large number (73) of better than memoryless optimal policies, 7 memoryless optimal, 4 optimal, and 1 other satisficing policy, the remaining 15 being non-satisficing. With $\alpha = 0.01$ both oracle agents learn significantly more optimal policies. The Action Oracle Agent learns 41 optimal policies, 10 better than memoryless optimal, 39 memoryless optimal, 3 other satisficing, and 7 non-satisficing. The State Oracle learns 31 optimal policies, 33 better than memoryless optimal, 30 memoryless optimal, 5 other satisficing, and 1 non-satisficing.

Over all, the plots for SARSA(λ) indicate that although the mean total steps for the three types of agent are close (Fig. 4), the actually policies that each learns varies significantly. Unlike the Eight Adjacent Squares Agent, both of the oracle agents learn optimal and better than memoryless optimal policies, especially when a lower value of α is used.

©Springer-Verlag http://www.springer.de/comp/lncs/index.html

6 Discussion & Conclusion

The locations where the oracles are consulted in Fig. 3 generally correspond to places where we would expect difficulties due to state aliasing. Thus it appears that our first hypothesis is confirmed, in that reinforcement learning algorithms can learn to make good use of external resources in order to clarify the agent's current state. This indicates that agents should be able to make use of active perception systems when solving partially observable navigation problems.

Our second hypothesis is also not falsified as either oracle agent when combined with SARSA(λ) and Watkins's Q(λ) can learn optimal policies. Plots for Watkins's Q(λ) are not shown; they are similar to SARSA(λ), although significantly fewer optimal policies are learnt.

The State Oracle Agent generates more optimal policies than the Action Oracle Agent for $\alpha = 0.1$. For $\alpha = 0.01$ it has a lower mean total steps than the Action Oracle Agent whilst almost matching the number of optimal solutions learnt by the Action Oracle Agent. The success of the State Oracle Agent is encouraging since compared to the Action Oracle, which has access to a known optimal policy, the State Oracle has no extra information about the task. The results indicate that in order to aid an agent dealing with a partially observable task, an active perception system only has to provide non-ambiguous representations (within the context of the task) for the current state, *i.e.* it does not have to provide any additional knowledge or reasoning about the problem.

The frequency with which optimal policies are learnt appears to be limited and dependent on the value of α. Possible causes of this limit could be: (i) the cost of perceptual actions is too high; (ii) that the advice given by the Action Oracle to its agent may be of little value unless the agent's own policy has converged to closely match that used by the Action Oracle; (iii) there is some step change in complexity between better than memoryless optimal and optimal policies for Sutton's Grid World.

We briefly looked at varying the cost of perceptual actions. As the perceptual action cost decreases, the number of optimum policies learnt by the Action Oracle Agent increases. However, as the perceptual cost tends to zero, the individual agents tend towards the rather uninteresting policy of always asking the Action Oracle what to do. Also, for the range of perceptual action rewards tried, the State Oracle Agent failed to learn any satisficing policies. There is no constraint preventing the State Oracle Agent from continuously requesting information from the State Oracle, and with the perceptual action cost reduced to -0.5, and a discount factor (γ) of 0.9, it is less costly to talk to the oracle forever, than to attempt to reach the goal from some of the more distant locations in the grid world. There is a limited cost range within which the State Oracle Agent should learn satisficing policies, however this range is task specific and identifying it requires prior knowledge of the problem. This we would prefer to avoid.

In conclusion, the oracles were introduced as an idealised active perception system, and their success suggests that the use of active perception should provide a feasible approach to solving partially observable navigation problems.

©Springer-Verlag http://www.springer.de/comp/lncs/index.html

7 Future Work

Future work flowing from this paper includes: (i) examining what prevents the combination of oracles and eligibility traces from generating a greater number of optimal solutions; (ii) generating comparative results for agents that use memory or internal models; (iii) extending these results to other grid world and also non-minimum time problems, *e.g.* McCallum's New York driving problem [10]; (iv) testing the performance of agents using actual active perception systems.

Acknowledgements. Paul Crook would like to thank EPSRC without whose funding this work would not be possible. We would also like to thank Alison Pease for proof reading drafts of this paper and the reviewers for their useful comments.

References

1. Paul A. Crook and Gillian Hayes. Learning in a state of confusion: Perceptual aliasing in grid world navigation. In *Towards Intelligent Mobile Robots 2003 (TIMR 2003), 4th British Conference on (Mobile) Robotics*, UWE, Bristol, 2003 *(in press)*.
2. Steven D. Whitehead. *Reinforcement Learning for the Adaptive Control of Perception and Action*. PhD thesis, University of Rochester, Department of Computer Science, Rochester, New York, 1992.
3. Pier Luca Lanzi. Adaptive agents with reinforcement learning and internal memory. In Jean-Arcady Meyer et al., editor, *From Animals to Animats 6: Proceedings of the Sixth International Conference on the Simulation of Adaptive Behavior (SAB'2000)*, pages 333–342. The MIT Press, Cambridge, MA, 2000.
4. Lonnie Chrisman. Reinforcement learning with perceptual aliasing: The perceptual distinctions approach. In *Tenth National Conference on Artificial Intelligence*, pages 183–188. AAAI/MIT Press, 1992.
5. Andrew Kachites McCallum. Overcoming incomplete perception with utile distinction memory. In *Tenth International Machine Learning Conference (ML'93)*, pages 190–196, Amherst, MA, 1993.
6. Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. The MIT Press, Cambridge, MA, 1998.
7. Michael L. Littman. Memoryless policies: Theoretical limitations and practical results. In Dave Cliff et al., editor, *From Animals to Animats 3: Proceedings of the Third International Conference on Simulation of Adaptive Behavior (SAB'94)*, pages 238–245. The MIT Press, Cambridge, MA, 1994.
8. Satinder P. Singh, Tommi Jaakkola, and Michael I. Jordan. Learning without state-estimation in partially observable Markovian decision processes. In *International Conference on Machine Learning*, pages 284–292, 1994.
9. John Loch and Satinder Singh. Using eligibility traces to find the best memoryless policy in partially observable Markov decision processes. In *Proc. 15th International Conf. on Machine Learning*, pages 323–331. Morgan Kaufmann, San Francisco, CA, 1998.
10. Andrew Kachites McCallum. Learning to use selective attention and short-term memory in sequential tasks. In P. Maes et al., editor, *From Animals to Animats 4: Proceedings of the Fourth International Conference on Simulation of Adaptive Behavior (SAB'96)*, pages 315–324. The MIT Press, Cambridge, MA, 1996.

©Springer-Verlag http://www.springer.de/comp/lncs/index.html