GENERALISATION, WORLD MODELLING, AND OPTIMAL CHOICE: IMPROVING REINFORCEMENT LEARNING IN REAL ROBOTS

O.E.Holland* and M.A.Snaith**

***Faculty of Engineering, University of the West of England, Coldharbour Lane, Frenchay, Bristol BS16 1QY, United Kingdom. email: owen@tag.co.uk**

****Technology Applications Group, 5, Bolams Mill, Dispensary Street, Alnwick, Northumberland NE66 1AN, United Kingdom. email: martin@tag.co.uk**

Abstract

This paper describes an autonomous mobile robot designed to learn to find its way through an arbitrary maze-like environment to a location at which it receives positive reinforcement, while avoiding locations at which it receives negative reinforcement. The robot is behaviour based, and includes behaviours for collision avoidance and crash recovery which enable it to move freely in unstructured environments and in the presence of dynamic obstacles. The robot represents its environmental situation in terms of one of a large number of possible discrete states defined by the current sensory input and the memory of the last few sensory inputs and output actions. The learning algorithm used is a variant of Q-learning; to maximise learning speed, action selection is deterministic. An enhancement is proposed to speed up learning by generalising across similar states. The potential gains offered by building and exploiting a deictic world model are considered, and an alternative to Sutton's Dyna is proposed.

.

1. INTRODUCTION

This paper describes the progress to date of an attempt to produce an autonomous mobile robot which, when placed in an arbitrary unstructured environment, will learn to find its way through that environment to goal locations at which it has received positive reward while avoiding locations at which it has received negative reward. The robot is required to avoid collisions with both dynamic and static obstacles, and is restricted to sensing only local features - there is no capacity for odometry or for sensing orientation. Before discussing the background to the technical aspects of the work, it will be useful to make some comments on the rationale behind this general approach and on its relevance to artificial life.

It is easy to divide the current practice of artificial life into two schools: one is concerned with exploring life-like phenomena in artificial worlds, and the other deals with the study of artificially produced real world systems which display some characteristics of life. The work reported here belongs to the second school, and to that strand of enquiry which is ultimately concerned with the manufacture of artificial creatures which behave (in some respects) as if they were alive. Unfortunately, the theoretical aspects of this have received little attention.

One reason for this is that the field of autonomous robotics grew out of artificial intelligence, and so the little theory that there is often merely reflects the historical concerns of AI in its struggle to cope with its failure to produce systems which deal competently with the real world. Debate about grounded symbols and so on is wholly appropriate for those whose concern is with the nature of intelligence, but it is important not to confuse intelligence with behaviour. Behaviour is simply what animals, including humans, are observed to do, in context. Zoologists and animal psychologists have found themselves able to study the behaviour of animals at all points on the phylogenetic scale without having to doubt that what they were studying was always animal behaviour, and without having to reduce that behaviour to the single dimension of intelligence in order to appreciate its significance. It would be very convenient for artificial life if the zoologists and animal psychologists had produced some objective criteria by which behaviour of animate origin might be characterised, because it might then be possible to claim that a robot produced behaviour with the same characteristics; however, they have not, although there were many attempts to do so during the early days of psychology (e.g McDougall's hormic theory).

Perhaps what is needed is a modified version of the Turing test. The observer would be able to observe the entity under test in its environment via some virtual reality interface which concealed or disguised only its physical appearance, and possibly the appearance of any structure or object having a motivational or metabolic significance for the entity. He would be permitted to interfere with any aspect of the environment, including the introduction and manipulation of any available objects of behavioural significance, and his task would be to infer from the entity's behaviour whether it was an animal or a robot. The observer may have to be constrained to make an entirely rational decision, rather than relying on his intuition, because one intriguing possibility is that humans may have some hard-wired perceptual system for the perception of animate agency. We are probably decades from having a machine worth testing in this way, but it is probably not too soon to begin to think about such a test, and the absence of a test does not affect the validity of attempts to produce lifelike behaviour in robots as part of the study of artificial life.

It is worth asking whether it is necessary to build and test a real robot as primitive as this. Would it not be at least as useful, and much more convenient, to test the

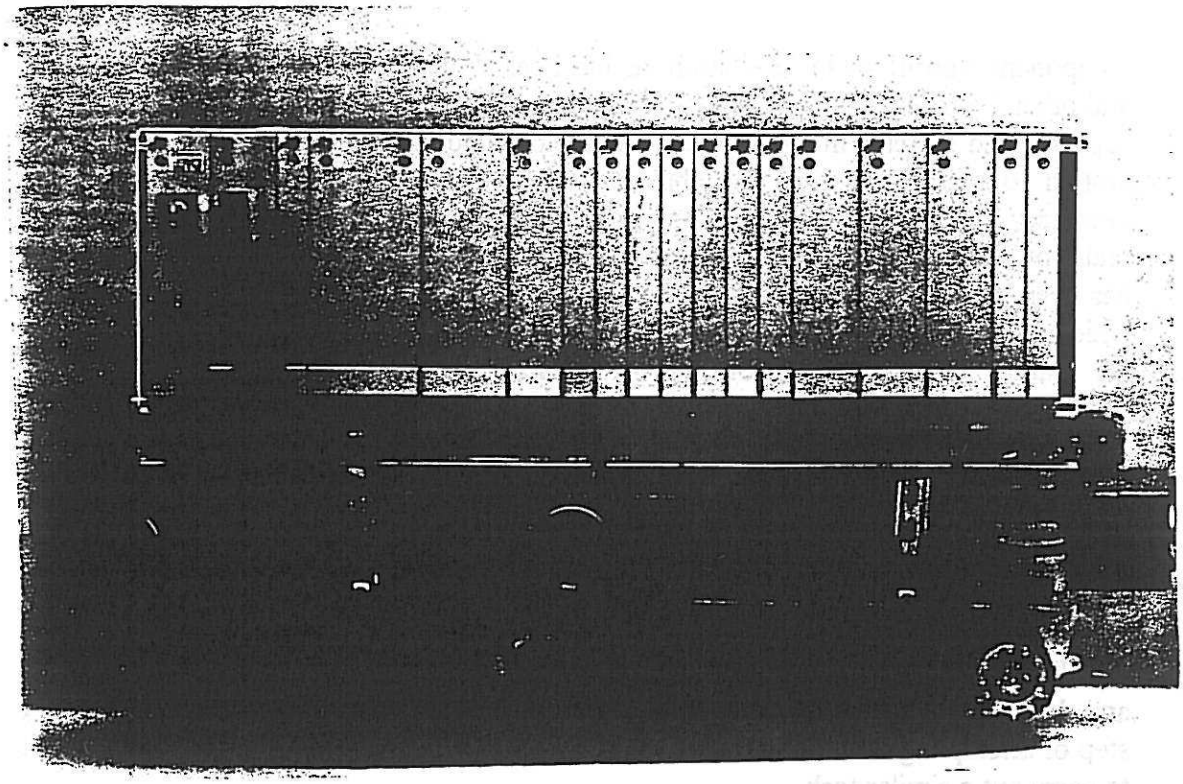
component algorithms in simulation, as has been done to great effect by Sutton, Lin, and others? Would progress not be made more rapidly through developing increasingly sophisticated systems in a computer-bound simulated universe, and simply porting the system to reality when it was able to do something worthwhile? or is it in fact necessary to develop the structural and functional elements of the robot bit by bit in the real world and arrive at the whole system by an incremental process? A similar debate has been running for some years now in the corner of the artificial intelligence world which is concerned with real and simulated robots; the main lesson so far has been that the nature of the real world [asynchronous, noisy, non-linear, unpredictable, continuous, and existing in inexorably real time] is so far different from the nature of any simulated world that the naive implementation on a real robot of an algorithm which appears to be successful in a simulation will generally end in failure. The pattern of the successes of recent years has followed the process of incremental growth through testing and development in the real world. Q-learning has already been shown to be useful for controlling goal-seeking navigation in simple simulated robots, which exist in a convenient time-discrete and space-discrete world, which can always unfailingly identify the grid-point they are on, and which do not have to avoid static and dynamic obstacles or recover from collisions; this work undertakes the necessary step of attempting to embed Q-learning into a real robot in the real world and to use it to carry out a similar task.

The behaviour-based techniques developed by Brooks and others (which increase overall competence incrementally by adding specific behaviours within the overall framework of subsumption-type architectures) are remarkably effective at dealing with some of the more intractable aspects of the real world. Some lines of argument have been developed which maintain that at least some functions in some animals are organised on a similar basis. From the hardware point of view, the modularity allows the development of implementations of behaviours which are flexible and robust, which greatly reduces the downtime of robots - undoubtedly the greatest problem of those working with real robots.

Reinforcement learning (providing the robot with no information about performance other than by giving reward and punishment) is used because it is presumed that animals have no other information available when they learn where to find food, water, and shelter, and to avoid physical discomfort, in a natural environment. It is also impossible in practice to know the sequence of output commands which a robot should produce at any time in order to reach the goal by some correct or optimal path, and so the feasibility of providing error-based feedback is questionable. Finally, attempting to provide error-based feedback would require either that the environment was suitably instrumented, or that a human trainer was present to supervise learning; these conflict with the ideas that the environment should be unstructured, and that the robot should be autonomous.

2. THE ROBOT

FRANK III is basically a 19 inch rack mounted on tracks, with a notebook PC mounted on top of the rack. The tracks may be driven independently at graded speeds forwards or backwards via PWM drivers; tachometers are fitted to the motors, and a speed servo is used. Three sensor podules are mounted at the front of the robot (orientated straight ahead, left, and right); each contains an ultrasonic range sensor, an active infra-red diffuse reflective sensor, and a two degree of freedom microswitch



FRANK, shown without the notebook computer

sensing deflection of the podule. A fourth podule is mounted at the rear of the robot, pointing backwards, and there are other infra-red sensors mounted around the periphery of the robot. The first few slots in the rack contain various items of housekeeping circuitry, and a DC-DC power converter which provides stabilised output voltages to the electronics and the computer, and which is powered by a single 12v lead-acid battery mounted in the chassis. The other slots allow a variety of boards to be plugged in to the 64-way backplane which also connects to the PC bus and to a fast multi-channel A-D and D-A card underneath the computer.

The robot architecture is a variant of Brooks' subsumption architecture. Behaviours implemented on plug-in cards may communicate with one another along the backplane, and may be arranged to suppress one another as required. However, instead of suppressing one another, behaviours may be configured to generate analogue output vectors for the motor drivers which are summed on the bus with other vectors; the resultant vector can produce useful behaviour fusion. At the lowest level, crash recovery behaviours (activated by the podule microswitches) suppress other behaviours, and produce a move away from the contact site, followed by a turn. Collision avoidance behaviours, triggered by the infra-red sensors, add a vector to the motor drivers tending to drive the robot away from the imminent collision. A pilot behaviour, driven by the ultrasonic range sensors, provides a default wandering behaviour, again in the form of a vector on the motor control bus. At the highest level, outputs from the computer are translated into analogue signals by the A-D card and appear as a vector on the motor control bus.

Simple behaviours are not implemented using microprocessors as in Brooks' robots; instead, customised assemblies of hardware analogue neuron models are used. A number of primitive neural functions have been identified (summation, inhibition, remanence, tolerance, threshold, and delay); these have been implemented in analogue circuitry on printed circuit boards (N-Eurocards) incorporating patchboards which

allow the functions to be connected in series and in parallel. Parameters (corresponding to synaptic strength, and to various time constants) can be set by potentiometers on the boards. Designs for a variety of behaviours have been produced over the last four years; because the designs tend to use predominantly series connections, they are referred to as neural strands. Installing a behaviour is simply a matter of configuring one or more N-Eurocards to the appropriate strands, plugging them in, and adjusting the parameters if necessary when the behaviour is tested. Parameter settings are robust,but are readily altered if a sensor output changes or if a sensor needs to be replaced.

The sensory information available to the learning system is severely restricted. The three ultrasonic rangefinders at the front of the robot (pointing left, right, and straight ahead) are used to derive eight primitive percepts, roughly defined by the presence or absence of a reflective surface in each of the three directions:

FREE_SPACE
WALL_AHEAD
CORNER_PORT
CORNER_STARBOARD
WALL_PORT
WALL_STARBOARD
CORRIDOR
TRAP

If a simple binary scheme had been used to generate these perceptions (with each sensor output switching at a predetermined range) the perceptions would have been liable to alternate rapidly at borderline positions, and would have been tied to fixed ranges. Instead, N-Eurocards are used, with eight strands, each producing an output reflecting the closeness of the input vector to the percept embodied in the strand. All eight strands are reciprocally interconnected with inhibitory connections to give a competitive network in which only one strand can produce a positive output. This arrangement (usually called an n-flop) also has intrinsic hysteresis. The net result is that perceptions are stable, do not alternate rapidly at borderlines, and exhibit some tolerance of changes in scale; the absolute strength of the winning perception is also available if required.

The output actions available to the learning system are also severely restricted. Four actions are available:

AHEAD
LEFT
RIGHT
STOP.

AHEAD corresponds to a vector commanding the left and right tracks to move at a certain speed; STOP corresponds to a vector requiring zero speed; LEFT corresponds to a vector requiring the left track to reverse and the right track to move forwards at a given speed; RIGHT is the opposite of LEFT. These actions are sustained - the robot will continue to turn on the spot or to advance for as long as the appropriate vector is being output. As all of these vectors will be combined with any other vectors on the bus at the time, the actual movement will not always correspond to the action output by the learning system; there is no feedback to the learning system to indicate the movement actually performed, although the information is in principle available from the bus and from the tacho outputs.

3. THE REINFORCEMENT LEARNING ALGORITHM

Q-learning, or incremental dynamic programming (Watkins, 1989; Sutton, 1990, 1991) is one of the simpler reinforcement learning architectures. In its basic form it requires a system with a finite number of states, and a finite number of actions available within each of those states. A scalar, the Q-value, is associated with each state-action pair. At each instant, the system selects one of the actions available in the current state; selection is usually probabilistic and is biased towards the action with the highest Q-value. The action will produce a transition to another state, and possibly a reward; the Q-value of the state-action pair just selected is then modified by the learning algorithm, which is a straightforward temporal difference type (Sutton, 1988). Proofs are available that, under certain circumstances, the Q-values in such a system will converge to the discounted returns of the state-action pairs, and so selection of the state-action pairs with the highest Q-values is an optimal strategy for maximising the long-term rate of acquisition of reward (Barto et al, 1989).

Q-learning has attracted a reputation for being slow. This may be because, in its pure form, each instance of reward (and these may be sparse when reward is associated only with a single goal state) propagates only one step back per trial, and so a solution involving at least n steps requires at least n trials. Unfortunately, to take only n trials, each such trial must reach the state which in the previous trial had received the relevant update, and in the absence of reward information this will have to be achieved by a random walk, which may be extremely long. Worse still, some variants of Q-learning include a bias towards exploration by making recently chosen actions less likely to be selected than actions not selected for some time; this makes a given state less likely to be encountered on temporally adjacent trials, and slows the learning process down further. However, there are a number of ways in which Q-learning can be accelerated (at least in simulation), such as the use of generalisation and world models, and so the slowness of Q-learning in its basic form does not necessarily make it a bad initial choice for the study of learning in real robots. It does have the great advantage of simplicity, which is a great help when attempting Braitenberg's uphill analysis after the exhilaration of the downhill synthesis.

In basic Q-learning, stochastic action selection is necessary until the Q-values have stabilised; after that, provided that the state transitions caused by the state-action pairs and the distribution of reward do not change, deterministic action selection is optimal. The stochastic action selection prevents the first action in a given state to acquire and maintain a positive Q-value from being the only action in that state ever to be selected subsequently. In this implementation, it was decided to make action selection stochastic if all Q-values in a given state were below some threshold, but deterministic otherwise. This decision grew out of the observation that, although in the long run optimality is clearly preferable, in the long run we are all dead, and a technique producing a relatively rapid approach to a reasonably good but rigid solution might represent a useful real-world compromise.

Q-values must clearly be initialised with some uniform or stochastically determined value. However, there are good arguments against the use of any particular value. Zero is attractive, but in systems with net positive return, it will understate the probable benefits of actions which have not yet been tried. A positive value higher than the average return will do the opposite. Stochastically determined values will add noise as well as bias. It might be possible to perform a rolling initialisation of all untried actions with the current mean value of all Q-values, but even this would introduce bias because Q-values are highest close to the goal, and so this would underestimate Q-values close

to the goal while overestimating them far from the goal. In this implementation, all Q-values are initialised with a character - the null character - which indicates to the algorithm that the associated actions have never been selected. An action is initialised with a value only when it has been selected; this allows a number of initialisation schemes to be tested.

If similar actions in similar states produce similar outcomes, it should be possible to estimate the Q-value of an untried state-action pair by using known Q-values from similar state-action pairs. Lin (1993) has used neural networks as the generalising algorithms in simulations of Q-learning, with some success. This implementation uses a cruder and quicker method: the estimated Q-value of an untried state-action pair is the average Q-value of the same action in states with some defined level of similarity to the current state (Holland and Snaith, 1992a) although this is not always an optimal use of the data (Holland and Snaith 1992b). If such an estimated value is the highest of any available action, the associated action will be selected. The update algorithm is then run using the estimated value as the original Q-value. This ensures that the only false information introduced into the system through initialisation is the difference between these estimated values and the 'true' values; it seems likely that this will be far less than that introduced by other methods of initialisation. It also avoids degrading the information represented in the Q-values, unlike Lin's method, and so Q-values should converge to the true return values faster. This method corresponds to the use of a reduced-state observer in conventional control theory - data points are left undisturbed.

Sutton (1990, 1991) has noted that both Q-learning and his Adaptive Heuristic Critic methods learn only what he calls 'what-to-do' (policy) and 'how-well-am-I-doing' (return predictions) but do not learn 'what-causes-what' (which he calls 'an internal model of the world's dynamics'). He has introduced extensions of both architectures (Dyna-AHC and Dyna-Q) which include an internal world model. The extra elements are structured to learn 'what-causes-what' which in Sutton's view is the information:
state x(t) + action a(t) -> reward r(t+1) + state x(t+1)
Sutton then randomly selects state-action pairs (sometimes biasing the selection to encourage exploration) and uses the world model to carry out a learning update without having to move the (simulated) robot. This achieves faster learning for a given number of simulated robot moves, and is very attractive when using a real robot.

It is possible to form a world model which contains exactly the same information but which is indexed by the result instead of the cause - in effect 'what-is-caused-by-what'. By using this information in a modified update procedure (Holland and Snaith, 1992c) which is triggered *only* when a real experience causes a significant change in a Q-value, and which recursively updates all experiences which could have led to the current experience according to the current world model, it should be possible to achieve even more efficient use of new information than by updating randomly chosen hypothetical experiences. There is of course some risk that a maze with a high branching factor could require a prohibitive amount of computation; this could be resolved by limiting the number of computations per update.

4. THE REPRESENTATION OF STATE AND TIME

The sensory information available to the robot is clearly utterly inadequate to disambiguate location in an unstructured environment; many locations will correspond to any one of the eight perceptions. However, by taking the present perception together with the previous action, the number of alternative locations supporting such a sequence will be fewer than those supporting the perception alone, and there will be

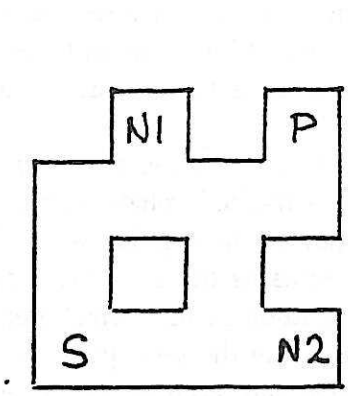
twenty-four distinct such sequences. By concatenating more successive perceptions and actions, the number of locations supporting a given sequence should reduce still further, and the number of distinct sequences will increase. This implementation uses the last five perceptions and the last five actions to define state, giving a theoretical maximum of almost 8 million distinct sequences, or PAS (Perception-Action-Sequence) encodings, which is thought to be sufficient to disambiguate enough locations in a normal room environment to permit navigation. However, in order to reduce the number of states for these initial experiments, the actions possible in any state are constrained by the current perception in such a way as to make them consistent with that perception. For example, when the current perception is CORRIDOR (objects to left and right, clear ahead) the only consistent action is FORWARDS. Where there is apparently only one consistent action, it has been found necessary to add a further option (STOP) because the unreliability of the ultrasound sensors means that there may in fact be some obstacle unsensed by the ultrasound but detectable at close range by active infra-red which will cause the only consistent action to be inhibited by the low-level avoidance circuitry linked to the infra-red, causing the robot to freeze; the STOP option allows the avoidance circuitry to operate unhindered and to move the robot sufficiently for the ultrasonic sensor to avoid the localised specular reflection and to give an accurate reading of range. In the case of TRAP (objects detected ahead, left, and right) LEFT and RIGHT were accepted as consistent actions. The theoretical maximum number of states is then reduced to 1,419,857.

In general, a new state is registered when the perception changes after an action, or when a time-out period (currently five seconds) has elapsed after the initiation of an action and no change of perception has occurred. The entry to a new state immediately terminates the action output by the previous state. Ther is one exception, which had be introduced during the initial series of experiments. When the robot turns, considerable force is exerted on the chassis, the tracks, and on the floor covering. When the turn is stopped, the relaxation of these elements often leads to a partial reversal of the turn. If a new perception occurred soon after the start of the turn, the robot could relax back into the starting position. In order to prevent this, the turns have a minimum duration, and changes of perception during this minimum duration do not cause a new state to be entered and the turn to be stopped.

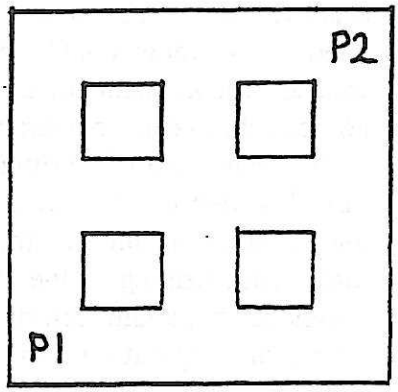
A simple and intuitively satisfactory measure of similarity is available between two states: it is defined as the number of consecutive elements they have in common including the current perception. It is easy to search across all the states with the last six elements the same as the current state, and to average the Q-values for each action requiring an estimated Q-value. However, the value of doing this is not yet known, and must be established from the data collected from this series of experiments. (Data from simulations could definitely prove useful here.)

5. THE ENVIRONMENT

In order to produce reproducible environments of graded complexity, it was decided to use maze-like structures made of cardboard and the rough side of hardboard - materials which give adequate diffuse reflection but which are not prone to producing multiple reflections. By using alleys which are too narrow for the robot to turn in, we can constrain the robot's paths to some extent, and can force it to arrive at various types of choice points. However, because of the binary nature of the collision avoidance sensors, the robot does does not run smoothly down the midline of the alleys, but weaves from side to side as the collision avoidance circuits kick in. The intrinsic unreliability of the ultrasonic sensors (which are vulnerable to multiple



Maze 1



Maze 2

reflections in a maze environment) can then produce 'incorrect' perceptions; the net effect is that there is some degree of sensitive dependence on the initial conditions. Rather than undertake the considerable task of changing the infra-red collision avoidance sensors to a rangefinding type, we have chosen to leave things as they are, reasoning that if the system cannot cope with this variability in a simple maze environment, then it is unlikely to be able to deal with a truly unstructured environment.

Reward may be delivered by manually pausing the robot and keying in the appropriate character on the notebook keyboard, or by using localised modulated infra red beacons in the mazes and a receiver on the robot. .

Maze 1 was used to evaluate the feasibility of the approach. Manual reward was used (negative at N1 and N2, positive at P) and the robot was removed from the maze after each reinforcement and restarted in a random orientation at S. In theory, the maze could be run in a minimum of eight steps, only two of which would involve any choice.

Maze 2 is currently being used for extended trials with full data recording. The robot is able to run for long periods with little or no user intervention. The sites P1 and P2 where positive reward is given (by beacons) are 'drive through' locations; there is no negative reward at present; there are more choice ponts, including a crossroads. The symmetry of the maze is exploited by arranging for each reward to set the reward detector to register only the other reward; a run from P1 to P2 is then effectively using the same maze as a run from P2 to P1, and so successive starts at P1 and P2 may be regarded as being successive trials on the same maze. Maze 2 can in theory be solved in a minimum of nine steps, three of which involve choice.

6. OBSERVATIONS AND RESULTS

In the initial trials the robot surprised us by consistently learning over the course of twenty or thirty trials to reach the goal state in around 15 steps. (We believe the practical minimum number of steps for this maze to be 13; this was achieved several times.) Performance often seemed inconsistent, in that a number of 'good' runs would be followed by a run showing little or no evidence of learning. However, performance

did seem to improve progressively as the number of trials was increased. Analysis of the state records generated by the first implementation again surprised us by showing that the total number of states experienced by a well-trained system in the maze was small (typically around 200). The generalisation mechanism was observed to work, altho the benefits could not be quantified. World modelling was not tested. The constant requirement for intervention limited the usefulness of this maze, and it was not feasible to carry out extended trials.

It was noticed while running the robot for long periods on Maze 1 that the distance turned tended to vary. Because of the way in which a tracked vehicle turns, much of this variability is due to variations in friction at the site of turning. However, frequent turns cause heating of the motor windings which decreases the available torque; this causes the minimum turn duration to result in a magnitude of turn which depends on the motor temperature, and so the same action command at the same point in the maze could produce different results. In order to control this completely, it would be necessary to use some closed loop control of the angle turned through, rather than a minimum duration; as a temporary measure, the motors have been fitted with a cooling fan, which has significantly reduced this variability.

Although apparently not much more complex than Maze 1, Maze 2 appears to present a much more difficult task to the robot; the reason for this is not yet understood. The number of states seen so far is much greater - the maximum is probably around five thousand - and learning, even with generalisation, is painfully slow in terms of real time. The robot can process a thousand steps (ten or twenty reinforcements) in three hours or so; weeks of work will be required to establish the asymptotic performance of the algorithm. Once this has been done, the experiment will be repeated using the world modelling algorithm. However, at the moment the indications are that the representation of position by the PAS encoding is far too weak, and future work will probably involve increasing sensor resolution and reliability, adding orientation and odometric information, and investigating a more effective generalisation mechanism.

8. REFERENCES

Barto A.G., Sutton R.S., and Watkins C.J.C.H. (1989). Learning and sequential decision making. COINS Technical report 89-95 University of Massachusetts at Amherst

Holland O.E. and Snaith M.A. (1992a). Q-learning with generalisation: an architecture for real-world reinforcement learning in a mobile robot. IJCNN-92, Baltimore

Holland O.E. and Snaith M.A. (1992b). When a bird in the bush is worth two in the hand. ICANN-92, Brighton

Holland O.E. and Snaith M.A. (1992c). Extending the Adaptive Heuristic Critic and Q-learning: from facts to implications. ICANN-92, Brighton

Long-Ji Lin (1993). Reinforcement learning for robots using neural networks. Ph.D. dissertation, School of Computer Science, Carnegie Mellon University, Pittsburgh

Sutton R.S. (1990). First results with Dyna. In: *Neural Networks for Control*,Ed. Miller W.T., Sutton R.S., and Werbos P.J., Bradford-MIT, Cambridge MA

Sutton, R.S. (1991). "Reinforcement Learning Architectures for Animats" In: *From Animals to Animats*, Ed. Meyer J-A and Wilson S.W., Bradford-MIT, Cambridge MA

Sutton, R.S. (1988). Learning to predict by the methods of temporal differences. *Machine Learning* 3, 9

Watkins, C.J.C.H. (1989). Learning with delayed rewards. Ph.D. Thesis, Cambridge University Psychology Department.