Autonomous Acquisition of Fuzzy Rules for Mobile Robot Control: First Results from two Evolutionary Computation Approaches

A. G. Pipe **B. Carse**

Intelligent Autonomous Systems engineering Laboratory, Faculty of Engineering,
University of the West of England, Frenchay Campus,
Coldharbour Lane, Bristol BS16 1QY, United Kingdom

Email: Anthony.Pipe@uwe.ac.uk
Web Site: http://www.ias.uwe.ac.uk
Phone: (+44) 117 3442818, FAX: (+44) 117 9763873

Abstract

We describe two architectures that autonomously acquire fuzzy control rules to provide reactive behavioural competencies in a simulated mobile robotics application. One architecture is a "Pittsburgh"-style Fuzzy Classifier System (Pitt1). The other architecture is a "Michigan"-style Fuzzy Classifier System (Mich1). We tested the architectures on their ability to acquire an "investigative" obstacle avoidance competency. We found that Mich1 implemented a more local incremental search than the other architecture. In simpler environments Mich1 was typically able to find adequate solutions with significantly fewer fitness evaluations. Since fitness evaluation can be very time consuming in this application, it could be a strong positive factor. However, when the rule set must implement a competency in more complex environments, the situation is somewhat different. The superior ability of Pitt1 to retain a number of schema in the population during the process of optimisation, is then a crucial strength.

1 BACKGROUND

Evolutionary Computation and Reinforcement Learning are both powerful techniques that can be utilised in creating entities capable of autonomously acquiring useful rules about a chosen problem domain. Well established approaches include those that use;

a) evolutionary techniques operating at the level of whole rule sets (Carse, Fogarty & Munro, 1996; Smith 1980),

b) evolutionary techniques that operate at the level of individual rules in a set (Booker, Goldberg & Holland, 1989),

c) other "lifetime" reinforcement learning approaches operating within a single rule set (Pipe, Fogarty & Winfield, 1994A, 1994B, 1996; Pipe & Carse, 1994; Pipe & Winfield, 1996; Sullivan & Pipe, 1996; Sutton, 1984).

Although each of the three categories listed above are at a quite mature stage within their own fields, the authors believe that a comparative investigation into the characteristics and performance of these techniques in some appropriate shared problem domain could be a very enlightening and fruitful area for research. We chose to conduct such a programme of work in the area of mobile robotics. This application area has characteristics that are complex but easy to visualise, it is widely known, it is a domain with which the authors have considerable experience, and the results of the research could have some future use in the real world. We have chosen fuzzy logic to implement local-cued behavioural control of a wheeled robot, the task therefore is to discover good fuzzy rules for implementing a particular competency in an artificial creature, or animat (Wilson, 1987). We have already conducted a considerable amount of work in the area of "lifetime" reinforcement learning applied to the application domain covered by this paper, i.e. category c) above. However, before conducting a thorough comparison we needed to adapt existing algorithms from categories a) and b) to our application domain. This paper therefore focuses on the structure of, and first tests on, an architecture drawn from each of these other two categories. In order to facilitate the future comparative studies between all three architectures, a common testing harness has been developed as part of the new work presented here; it includes the environmental and robot simulations, as well as the fuzzy logic system that controls the robot. In order to allow the experiments to be ratified, and perhaps extended, by others – all of this test harness software is available by email or by visiting our web site, both addresses are given at the head of this paper.

Before going further it is worthwhile briefly reviewing our existing "lifetime" reinforcement learning architecture, that lies in category c), to help set the scene for this paper. When used for the purpose of extracting local-cued fuzzy rules, it is a two-stage process. The first stage is goal-oriented. An Adaptive Heuristic Critic (AHC) architecture builds a "potential field" spatial cognitive map of an environment . It investigates the environment in a step-wise fashion by using an Evolution strategy (Rechenberg, 1973) to optimise the next move from the current position, until some goal location is reached. As it interacts with the environment it stores knowledge gained about "good" and "bad" places in a Radial Basis Function neural network. This neural network encodes the "potential field" map. From this map a number of useful trajectories through the environment can be autonomously created. This ends the first stage. During traversals of these trajectories sensorimotor data can be stored. A fuzzy clustering approach is then used to form local-cued rules from this data, and thus a behavioural module is created that encapsulates some competency (Pipe, Fogarty & Winfield, 1996; Pipe & Winfield, 1996). A number of competencies have been extracted in this way, including the obstacle avoidance competency that is the subject of this paper. This architecture is at a quite mature stage of development, it is currently being transferred to real-world domains.

However, it is clear that this is a quite complex multi-purpose architecture. Amongst our other aims, we wished to investigate methods for extracting fuzzy rules from environmental experiences in a more direct fashion. The focus of this paper therefore is on describing two new architectures that acquire fuzzy rules as a single stage process, and the first results of applying them in this common domain. Tuning of the evolutionary operators and other parameters for these two architectures must be the next step, followed by extensive comparative performance testing of all three architectures; but this is future work.

2 THE APPLICATION

It is clear from studies in the natural domain that many creatures make use of conscious and sub-conscious cognitive processing for reasoning about the future outcome of planned actions in the environment. Our work on cognitive map-building architectures, that work at this level, has been published elsewhere. However, such levels of information processing are not the focus of this paper. The vast majority of real (and artificial) creatures must also make extensive use of unconscious fast reactive behaviour, especially in the face of a volatile environment; in fact these abilities are often crucial for survival in a large range of circumstances. In mobile robotics, this level of control has been one of the focuses of attention in the Adaptive Behaviour research community for some time, and in particular Behaviour-Based robotics (Brooks, 1986). It seems clear from recent studies that whilst some reactive behaviours may require "internal state, or *weak internal representations* (Clark & Grush, 1999; Clark & Wheeler, 1998), many others are purely Stimulus-Response (S-R) and are used to good effect in both natural and artificial systems.

This paper begins the planned comparative work by making initial investigations into the abilities of the two new architectures to extract a useful S-R behavioural module from environmental experiences. Such a module must encapsulate an environmentally reactive competency. Examples are obstacle avoidance, taking right or left turns at a corridor T-junction, and so on. We have chosen an "investigative" obstacle avoidance competency for these first experiments. Because the behaviours are to be S-R at this stage of the work, any linkages between rules are made via the environment itself; there is no need to build internally linked behavioural sequences, and therefore the optimisation and/or learning tasks are simplified. We have used robot and environmental simulations extensively in our other previous research, the current new test harness design has built on these experiences, and is based heavily in real robot experimentation carried out in our laboratory. Details of the harness are given briefly below. However, as mentioned earlier, the C source code is freely available on request to the email address or directly from our laboratory's web site.

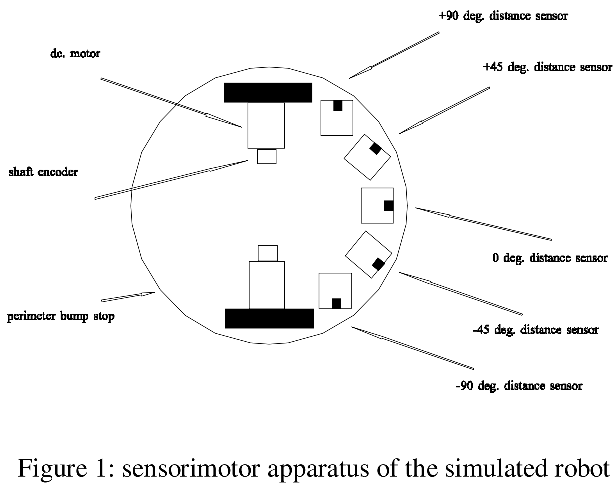

Figure 1: sensorimotor apparatus of the simulated robot

2.1 THE SIMULATED ROBOT

The following is a general description of the simulated twin-wheeled differential drive robot and its sensorimotor apparatus, illustrated in figure 1. The real robots in our laboratory possess two geared d.c. motors with an incremental shaft encoder on each. They are used in a low-level feedback loop to provide position and velocity control. These controllers are coupled through a kinematic algorithm to give a body-centred “virtual steering wheel”. The simulated environment therefore assumes that such a low-level control system is present, allowing control to be effected by an equivalent steering angle and forward velocity. In this work the robot travels through its environment with a constant forward speed of 0.1 m/s and a maximum continuously variable turning speed of 0.5 rad/s. The robot has an array of five distance sensors. The simulation supports a simple point-to-point measurement, to which noise and bias errors may be added if required, these are based upon ultrasonic sensors used on our real robots. The set of distance measuring sensors form a five element array, set at the following angles from the “straight ahead” position; 0^o, 90^o to the left, 45^o to the left, 45^o to the right, and 90^o to the right, each with a 5 metre maximum sensing range and intended for obtaining a local-cued environmental “signature”. A fuller description of the kinematic details used to generate the simulation of movement and of the type of distance sensors are also available via email address or at our web site.

2.2 THE SIMULATED ENVIRONMENT

The environmental mazes are set on rectangles of any size, although for the experiments reported in this paper they are square, being either 4metres or 10metres on each side. Any number of rectangular obstacles, of any dimension, may be placed in a maze. If there are start and goal positions, they may also be placed anywhere. It should be stressed that choosing rectangular shapes for the obstacles and the maze was purely an expedient in generating the maze simulation. The animat itself has no such restrictions in its sensory or motor parts. All measurements made and movements executed by the robot are continuous real valued, so for this simulation there is no concept of a “grid” or discretised state space.

When operated in normal mode, simulated animats sense and act in real time; for example velocities and sensory sampling intervals, established from observing actual vehicles in the laboratory, are tied to a real time clock with a period of 100ms. The simulation may also be operated in “super” real time. Under these circumstances 100ms of simulated time equates to approximately 10ms of real time on our 300MHz Pentium II PC, but the speed-up factor is then computer dependent. All other characteristics are, however, unaffected. This feature is useful since this simulation generates fitness evaluations for the two architectures tested and, like many evolutionary computation applications, it is fitness evaluation that is the primary time consuming factor.

2.3 IMPLEMENTING LOCAL-CUED S-R BEHAVIOURS USING FUZZY LOGIC

In the work presented in this paper, the fuzzy membership functions are fixed beforehand for both the input and output spaces, rule acquisition is limited to the creation and deletion of rules. When active as the robot's controller the Fuzzy Logic System (FLS) is run through one forward pass every 100ms simulation clock cycle, providing an updated steering angle for that period. The fuzzy controller has five inputs, one from each of the distance sensors and a single output defining steering angle. If fuzzy rule strength falls below a minimum threshold, then motion continues on a “straight-ahead” setting, so that minimally active rules are not able to influence the steering control.

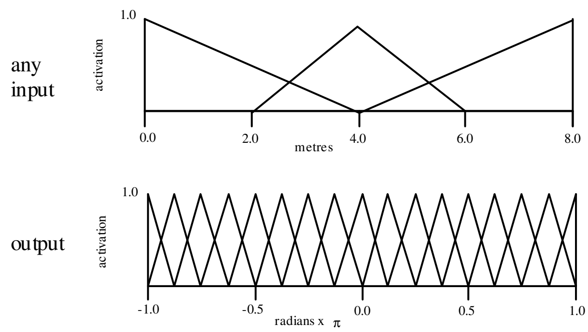

Figure 2: fuzzy membership function distributions

The FLS is a “Mamdani”-style system (Mamdani & Assilian, 1975). A conventional distribution of unit-height triangular membership functions was chosen. All functions were identical and equally spaced, with the exception of each function placed at the end of the range of an input or output, as shown in figure 2. For fuzzy AND a product of membership function activations was used for a given rule as opposed to the simpler MIN operator, since it requires little extra processing and is known to produce superior interpolation properties (Harris, 1992). Defuzzification was performed by conventional centre of gravity calculations. The use of 3 membership functions at each input and 17 at the output was established during previous research as being appropriate for this type of fuzzy controller in this application (Pipe, Fogarty & Winfield, 1996; Pipe & Winfield, 1996) and incorporated into this test harness. The reasons for choosing these parameters are given in that paper.

Table 1: format for a fuzzy rule

0	45L	90L	45R	90R	OUT

Each fuzzy rule was of the form shown in table 1, where; each of the six fields is an integer specifying a fuzzy membership function to use for that input or the output in forming a rule - counting from left to right on each graph shown in figure 2 (i.e. the interval (1-3) for each input and (1-17) for the output),

0	Membership Function (MF) number to use for front pointing distance sensor,
45L	MF number to use for distance sensor pointing 45^0 to the left of front,
90L	MF number to use for distance sensor pointing 90^0 to the left of front,
45R	MF number to use for distance sensor pointing 45^0 to the right of front,
90R	MF number to use for distance sensor pointing 90^0 to the right of front,
OUT	MF number to use for output angle field in radians x π – where positive values indicate a clockwise turning angle from the current orientation

All rules must specify a membership function for each input, i.e. in these first versions of the algorithms no "don't care" symbol is used.

Table 2: Details of Pitt1

Rule set size	50
Population size (no. rule sets)	40
Crossover operator	Single point, restricted to valid split between one complete rule and another in a rule set. Probability = 0.9.
Mutation operator	Two-point, one in a randomly selected input for a rule and changing its value to a randomly selected new membership function for that input, the other randomly selecting any valid new output membership function for the same selected rule with equal likelihood. Probability = 0.01, evaluated separately for each point.
Selection operator	"Roulette" wheel with "elitism", i.e. the highest fitness population member is retained in the subsequent generation, but with fitness re-evaluated. Each selection identifies two parent rule sets used to create a single offspring.

3 USING A "PITTSBURGH"-STYLE FUZZY CLASSIFIER SYSTEM

An evolutionary algorithm operating at this population based level, is analogous to the well known natural processes of evolution. In this early work, a "vanilla" GA was used to encode a population of fixed-size rule sets. The rule sets are evaluated for fitness by running a trial of the animat through a chosen simulated environment for each rule set in the population. When all rule sets have been evaluated in this way, the GA applies its operators to produce the next generation. This carries on until the process is halted by the designer, or the maximum number of generations is reached. Table 2 shows the details of operators, fitness function, and parameter values used.

Table 3: Details of Mich1

Population size (no. of rules in rule set)	200
Number of parent rules	50
Crossover operator	Two point, one at a randomly selected boundary between two input, the other selecting one or the other output MF identifier from the two selected parents with equal likelihood. Probability = 0.9.
Mutation operator	Two-point, one in a randomly selected input for a rule and changing its value to a randomly selected new membership function for that input, the other randomly selecting any valid new output membership function for the same selected rule with equal likelihood. Probability = 0.01, evaluated separately for each point.
Selection operator	"Roulette" wheel with a form of "elitism", i.e. all parent rules retained in the subsequent generation, but with fitness re-evaluated. Each selection identifies two parent rules used to create a single offspring.

4 USING A "MICHIGAN"-STYLE FUZZY CLASSIFIER SYSTEM

In our "Michigan"-style approach to this problem, an evolutionary algorithm acts upon some subset of a single set of rules. The elements of the evolutionary algorithm's population are therefore rules of a single rule set, rather than a group of rule sets as in the previous architecture. Again, for this early work, a simple system was created. A "vanilla" GA applies its operators to create a new single rule set at each generation. A group of the highest fitness scoring rules are used as parents for creating a new generation. An "elitism" operator retains this group into that next generation, but with fitness re-evaluated at that time. Fitness evaluation also now operates at the level of individual rules, carried out during a single simulation trial of the animat in a maze. Each rule's fitness is evaluated during this trial, the GA then produces the next generation, and so on. Table 3 shows the details for this implementation.

5 EXPERIMENTS

As mentioned earlier, in all of the experiments below, the acquisition of an obstacle avoidance competency was the overall aim. Of course for such a competency to be useful, investigation of the environment must be encouraged, otherwise a stationary animat could be deemed highly fit for the purpose. Therefore the fitness functions for both architectures included a measure of the maximum "straight-line" distance travelled by the animat from the start location during a trial in the environment. In fact just this single factor was adequate to form the fitness function for Pitt1.

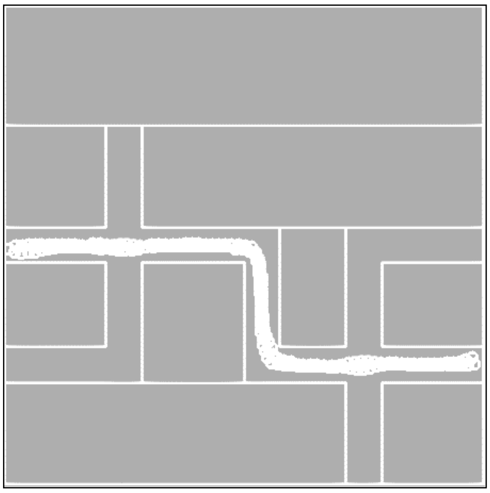

Figure 3: best Pitt1 member at generation 7

However, in the case of Mich1, each rule had to have its own fitness value, and therefore an additional factor related to cumulative activation of the rule was included. Since this is computed as part of the fuzzy inference process anyway, this did not incur a significant processing overhead; the rule activity was simply accumulated over the trial and then multiplied by the same "distance travelled" factor used as the fitness function for the other architecture. Any environmental trial was terminated under either of two conditions, if a maximum time allocation of 200 simulated seconds was reached, or there was an environmental collision before this time.

Many experiments were conducted, a typical sample is described below. In all of these examples good, and interesting, individuals were created before the maximum number of GA generations was reached, and particular examples like this are used in the discussion because they possess some feature that is noteworthy. However in approximately 80% of the experiments conducted to date on both architectures the population would converge on a good solution, if the parameters were set to those given in tables 2 & 3. In the other 20% of cases the population converged on a good solution, and then diverged again, with insufficient generations remaining for re-convergence.

We started experiments with a "corridor"-style 4metre square maze, as shown in figure 3. Either of the two architectures were able to form good obstacle avoiding trajectories through this environment within 20 generations. Even here though, the superior global search of Pitt1 found adequate solutions in fewer generations, this trend was more exaggerated in later experiments. Figure 3 shows a robot trajectory that begins on the left of the figure, but it is actually 5 successive traversals overlaid on each other. The simulation was terminated at this point for reasons of clarity, but there is no reason to believe that it would not continue indefinitely. This rule set was discovered at generation 7, but there was an adequate solution in the initial randomly selected population for this run, and this was not untypical for this architecture and this maze.

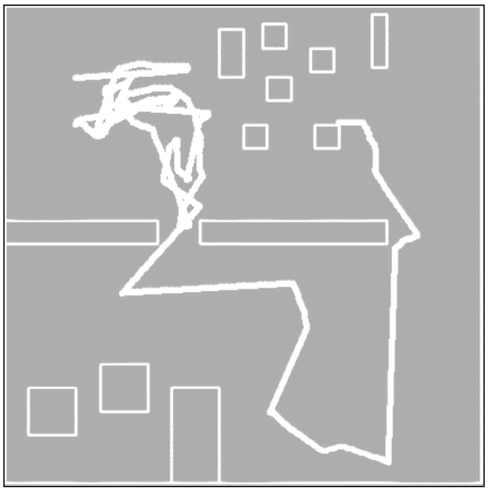

Figure 4: best Pitt1 member at generation 4

However this superior global search is not without penalty. Each generation, Pitt1 runs a robot through the maze 50 times in order to evaluate the fitness of each rule set. Mich1, by comparison, only runs a single robot trial per generation. If "super" real time simulation were to remain useful for real world applications, then this will not be a significant penalty for Pitt1. However, if there was a requirement for some significant number of evaluations to be carried out on a real robot, then this could be a highly negative aspect. In fact the "about-turn" repetitive nature of the trajectory that was created in this run was a side-product of evolution, since fitness is calculated from the furthest "straight-line" distance travelled to, at any point in a trial. For both architectures, although there were normally less than 5 rules with large turning angles as output in the best individual, typically

20 to 50 rules became active at some point in a trial. These other rules were responsible for maintaining stability during gradual turns; their removal soon revealed this fact.

A harder maze, shown in figure 4, was used for the remainder of the experiments reported on in this paper. This was a bigger, 10metre square maze; it was meant to represent a "warehouse"-style environment. There is an area at the top-right that is quite closed-in, a pair of parallel walls in the centre with two smaller openings, a collection of larger objects at the bottom-left, and some quite large open spaces. The reason that this environment was harder for these architectures to learn obstacle-avoiding rules for, lies in the differing characteristics of the sections in the environment. For example, different categories of rules are required in the closed in areas, compared with the open spaces. This means that the architectures must be more efficient in storing knowledge in their rule sets, in order to have adequate behavioural coverage within the same size of fuzzy logic system as the previous experiments. Figure 4 also shows a trajectory generated by the best rule set at generation 4 of Pitt1. The trajectory begins at the top left. It starts with some general avoidance behaviour in the upper-left quadrant. After two unsuccessful attempts, it then passes through one of the "doorways" in the central wall section. In fact there were some quite fit individuals in the first generation of this run, but it took until generation 4 for avoidance manoeuvring to emerge in the bottom-right quadrant.

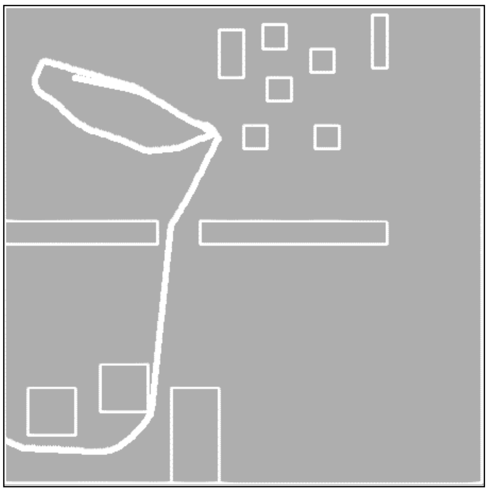

Figure 5: Mich1 at generation 37

Figure 5 shows an example trajectory generated by Mich1 at generation 37, with the same starting position. It is clear from figures 4 and 5 that in neither case had the rules acquired the ability to deal with closed-in environments. Both of them terminated before the maximum time allocation was reached, after collision in one of the closed in areas. In fact this problem was never surmounted despite 20 repetitions of these experiments on each architecture. However the nature of the fitness function did not encourage the creation of "close quarters" rules under these circumstances, both of the controllers illustrated in figures 4 and 5 were highly fit individuals.

However, simply changing the start location so that it was within the closed in section at the top-right of the maze effected a crucial change in the learning experience. The architectures were forced to create "close-quarters" rules in order to escape into the more open sections, and thereby earn high fitness values. This proved to be quite a lot harder for Mich1 than Pitt1.

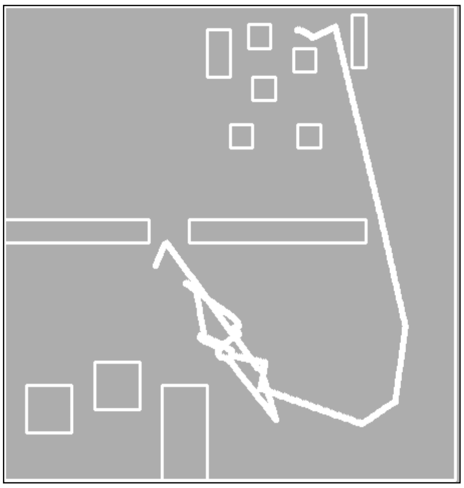

Figure 6: best Pitt1 member at generation 8

Figure 6 shows a trajectory generated by the best individual at only generation 8 of Pitt1. In subsequent generations the population converged on a slight variant of the best generation 8 individual. The trajectory stops in the mid-left of the figure because the maximum time allocation was reached; when this was doubled the animat continued move around in the environment for that increased time.

Figure 7 shows an example of Mich1 at generation 44. In this run performance did not improve significantly over the next 156 generations, and this was typical for this architecture when presented with this kind of problem. In fact there were only 16 rules that were active for significant amounts of time during this trial. There was a set that caused the early cycling behaviour close to the start location, and a second set that were responsible for the sharp turns. During the long straight runs no rules were active beyond a very minimal level, and were all below the cut-off threshold for steering control of the robot. As mentioned earlier, the result is "straight-ahead" motion at the standard speed until either one or more rules becomes active at a level greater than the threshold, or collision occurs.

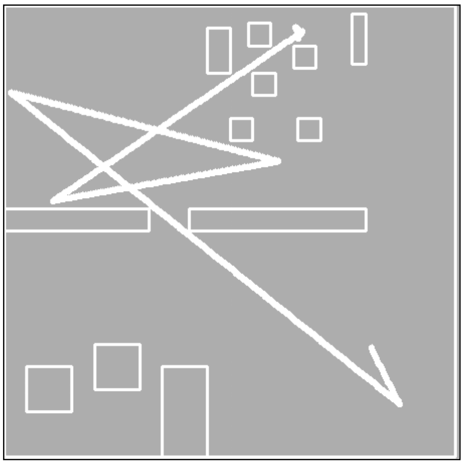

Figure 7: Mich1 at generation 44

The values of the parameters for each algorithm were determined experimentally. More work could be carried out in this area, but some indicative understanding has already been gained. The effects of changing parameter values of Pitt1 is described first. Reducing the rule set size caused incremental reduction in convergence performance and quality of the best individual. Increasing it did not yield linear improvements in performance of the algorithm, indicating that, for these problems, there was a "knee" characteristic to the curve of performance against rule set size. Decreasing the population size down to just 10 individuals still allowed sporadic convergence on good performance rules sets, whilst quadrupling this parameter value to 200 did not seem to give significantly improved performance. The effects of changing crossover and mutation probabilities were concomitant with what one would expect from a GA in most circumstances. Let us now consider changing parameters in Mich1. Reducing the number of rules used as parents for the next generation made convergence performance more volatile across repeated trials, doubling this number did not give significant improvement, and tended to increase the likelihood of premature sub-optimal convergence. Halving the population size was detrimental to performance in all cases, there did not seem to be enough coverage in the initial population under these circumstances. Doubling this parameter did slightly improve performance in the "initially closed in warehouse" environment, but gave significantly less than linear improvement. Like the other architecture, changing crossover and mutation probabilities yielded unsurprising results.

6 DISCUSSION

For these architectures, and in this application, Mich1 displayed a more incremental learning tendency than Pitt1. Increasing the size of the single rule set beyond that presented here only reduced, rather than removing, this tendency. Where the problem suited this methodology however, the benefits of the reduced number of time consuming fitness evaluations was a heavy factor in Mich1's favour. However, Pitt1's ability to retain the schema required for more complex problems as the optimisation process proceeded, gave it a distinct advantage under more complex circumstances.

These differences were best illustrated by the seemingly simple change of start location in the "warehouse" maze. If the animat was started in an open space, then only "open space obstacle avoidance" behaviour needed to be developed to create quite high fitness rule-bases. However, when started in a constrained space, rules for handling obstacle avoidance under these circumstances had to be developed simultaneously with those for open space scenarios.

In the case of the behaviour displayed in figure 7, one could conclude that Mich1 had successfully taken advantage of the combined features of the robot, the environment and fitness function, to create a fit individual with a very small number of useful rules. Perhaps this would be difficult to argue with under appropriate circumstance, after all if the only aim was to create an animat that investigates its environment, then this fuzzy controller achieves that approximately as well as the alternative approach illustrated in figure 6. However, it must be said that the control effort displayed in figure 7 looks rather brittle and inelegant.

7 FURTHER WORK

This work is clearly at an early stage and their is much more to be done. The performance of the two architectures displays promise, but they are both simple "vanilla" algorithms. To date, they have only been applied to a quite simple reactive behavioural problem, as shown by the small number of generations required for Pitt1 to find adequate solutions.

The next step must therefore be to bring the two architectures to a more mature state. To achieve this we must test a wide range of parameter values in a more rigorous fashion than has been conducted to date, and alternative operators must be investigated. In this way a firmer understanding may be reached of how to achieve internal optimality and robustness of the two underlying algorithms considered here.

In addition to these general points, there are some more specific problems. Firstly, we must test these two new architectures on their ability to extract other competencies. For many competencies this will require a solution to the problem of building internally linked behavioural sequences; i.e. a fuzzy rule-base that is able to encapsulate internal state, or *weak internal representations*, in some form when the problem requires it. From the evolutionary computation perspective, it is never simple to devise robust evolutionary operators that respect run-time created boundaries around groups of population members. Similarly, from the reinforcement learning perspective, Temporal Difference learning algorithms that are capable of dealing with linked chains of behaviour that eventually lead to positive reward, are also far from non-trivial in complexity. For Pitt1 this will mean that operators will have to be brought to bear that tend not to break-up internally linked rule groups. This could be achieved by converting the architecture into a Delayed Action Classifier System (Carse & Fogarty, 1994). For Mich1 a Temporal Difference delayed reward algorithm will need to be introduced (Sutton, 1984). Secondly, the form of the rules themselves is very restrictive to building a hierarchy of rules in a rule set. For example, the requirement for every rule to use all five inputs precludes the creation of a sub-group of "general" rules that use some critical subset of the inputs to provide powerful drive outputs; fuzzy logic, as a wider field, is full of such practices. Thirdly, only allowing each architecture to operate on the rules of the fuzzy logic system could also be restrictive. Membership function positions, and perhaps shapes, could also be part of the process.

8 CONCLUSIONS

The objectives of the work presented in this paper were to introduce the main aims of the wider programme of comparative work, describe the two new architectures that will be used in this programme, demonstrate them in operation in their preliminary form, and give some general indications as to their inherent strengths and weaknesses. In these respects we believe that the paper has served its purpose.

References

Booker L B, Goldberg D E & Holland J H (1989) Classifier Systems and Genetic Algorithms, AI 40, pp.235-282

Brooks R A (1986) A Robust Layered Control System for a Mobile Robot, IEEE J. of Robotics and Auto., RA-2, no. 1

Carse B & Fogarty T C (1994) A Delayed-Action Classifier System for Learning in Temporal Environments, Procs. IEEE Int. Symp. on Evolutionary Computation, Vol. 2 pp 670-673, Florida, USA

Carse B, Fogarty T C & Munro A (1996) Evolving fuzzy rule based controllers using genetic algorithms, Fuzzy Sets and Systems 80, pp.273-293

Clark A & Grush R (1999) Towards a Cognitive Robotics, Journal of Adaptive Behavior, 7 (1), International Society for Adaptive Behavior, pp.5-16

Clark A & Wheeler M (1998) Bringing Representation Back to Life, From Animals to Animats 5, Proceedings of fifth International Conference on Simulation of Adaptive Behavior, pp.3-12

Harris C J (1992) Comparative aspects of neural networks and fuzzy logic for real time control, in Neural Networks for Control and Systems, IEE Control Eng. Series #46, chap. 5, Peter Peregrinus, pp.72-93

Holland J H (1975) Adaptation in natural and artificial systems, University of Michigan Press, Ann Arbor

Mamdani E H & Assilian S (1975) An experiment in linguistic synthesis with a fuzzy logic controller. International Journal of Man-Machine Studies, vol. 7, no. 1, pp.1-13

Pipe A G, Fogarty T C & Winfield A (1994A) Hybrid Adaptive Heuristic Critic Architectures for Learning in Mazes with Continuous Search Spaces, Lecture Notes in Computer Science #866 Parallel Problem Solving from Nature III, Springer-Verlag, Eds. Davidor, Schwefel, manner, pp 482-491

Pipe A G, Fogarty T C & Winfield A (1994B) A Hybrid Architecture for Learning in Continuous Environmental Models in Maze Problems, From Animals to Animats 3, Ed. Cliff, Husbands, Meyer & Wilson, MIT Press, pp. 198-205

Pipe A G & Carse B (1994) A Comparison between two Architectures for Searching & Learning in Maze Problems, Lecture Notes Comp. Science #865, Springer-Verlag, Ed. T C Fogarty, pp.238-249

Pipe A G, Fogarty T C & Winfield A (1996) An Experiment in Knowledge Abstraction - From Cognitive Maps to Behaviours, Procs. 2nd Int. Conf. on Adaptive Computing in Engineering Design and Control, pp.65-70

Pipe A G & Winfield A (1996) An Autonomous System for Extracting Fuzzy Behavioural Rules in Mobile Robotics, From Animals to Animats 4, Cape Cod, USA, MIT Press, ISBN 0-262-63178-4, pp.233-244

Rechenberg I (1973) Evolutionsstrategie, Problemata, Fromman-Holzboog, Stuttgart, Germany

Smith S F (1980) A learning system based on genetic adaptive algorithms, PhD thesis, University of Pittsburgh

Sullivan J C W & Pipe A G (1996) Efficient Evolution Strategies for Exploration in Mobile Robotics, Procs. AISB workshop of Evolutionary Computation, Lecture Notes in Computing #1143, Springer Verlag, pp.147-161

Sutton R S (1984) PhD thesis `Temporal Credit Assignment in Reinforcement Learning', University of Massachusetts, Dept. of computer and Information Science

Wilson S W (1987) Classifier Systems and the Animat Problem. Machine Learning 2 (3), pp.199-228