Evolution versus Design: Controlling Autonomous Robots

Philip Husbands and Inman Harvey
School of Cognitive and Computing Sciences
University of Sussex
Brighton BN1 9QH, England
email: philh@cogs.susx.ac.uk inmanh@cogs.susx.ac.uk

Abstract

This paper sets out and justifies a methodology for the development of the control systems, or 'cognitive architectures', of autonomous mobile robots. It will be argued that the design by hand of such control systems becomes prohibitively difficult as complexity increases.

The alternative approach of artificial evolution is presented. It is argued that the most useful basic building blocks for an evolved cognitive architecture are adaptive noise tolerant neural networks rather than programs. These networks may be recurrent, and should operate in real time. Evolution should be incremental, using an extended and modified version of genetic algorithms.

Time constraints mean that architecture evaluations must be largely done in simulation. Results from a simulation are presented. The pitfalls of simulations compared with reality is discussed, together with the importance of incorporating noise.

1 Introduction

This paper sets out and justifies a methodology for the development of the control systems, or 'cognitive architectures', of autonomous mobile robots intended for use in unstructured or dynamic environments. Such robots will require active perception and will be behaviour-based. Although behaviour-based approaches to robot control appear to be far more promising than traditional model-based functional decomposition methods, it will be argued that the design of such control systems is still prohibitively difficult. A methodology based on the alternative approach of artificial evolution is presented. The advantages of such a scheme over design by hand are described.

The methodology is further illuminated by describing its application to the development of the cognitive architecture of a mobile autonomous robot engaged in a series of increasingly complex tasks. The robot is equipped with low resolution sensing devices and is required to act in uncertain environments. It is argued, in this case, that the most useful basic building blocks for an evolved cognitive architecture are adaptive noise tolerant neural networks.

Relevant work by others will be discussed. There have recently independently been suggestions of different but related evolutionary approaches. In particular during the preparation of this paper Brooks proposed using Evolutionary Programming techniques [8]; and Beer and Gallagher reported on the use of dynamical neural networks [5]. These methods will be compared with ours.

Artificial evolution requires the evaluation of a great number of candidate control systems. Time constraints mean that many of these evaluations must be done in simulation. The requirements of such a simulation system are discussed.

2 Interesting robots are too difficult to design

Traditional approaches to the development of autonomous robot control systems have made only modest progress, with fragile and computationally very expensive methods. A large part of the blame for this can be laid at the feet of an implicit assumption of functional decomposition — in the terms of the themes of this conference, the assumption that perception, planning and action could be analysed independently of each other. This failure has led to recent work at MIT which bases robot control architectures instead around *behavioural decomposition*[6, 7]. Such work rejects the traditional AI approach which manipulates symbolic representations of the world, and places more emphasis on 'knowing how' to do things rather than 'knowing that' the world is in a given state. Viewpoints sympathetic to such an approach can be seen in, e.g., [22, 4, 10, 2].

Such a subsumption-style cognitive architecture for

0-8186-2675-5/92 $3.00 © 1992 IEEE

a robot in theory analyses independent behaviours of a robot or *animat*,[1] such that each behaviour can be 'wired in' all the way from sensor input to motor output. Simple behaviours are wired in at first, and then more complex behaviours are added as a separate layer, affecting earlier layers only by means of suppression or inhibition mechanisms.

However it is accepted that the design of robust mobile robot control systems is highly complex because of the extreme difficulty of foreseeing all possible interactions with the environment; and the interactions between separate parts of the robot itself [7, 18]. The design by hand of such a cognitive architectures inherently becomes more complex much faster than the number of layers or modules within the architecture — the complexity can scale with the number of possible interactions between modules.

One way out of this problem to try and automate the design process. A possible approach is to view the design of a control architecture as a planning problem. Traditional AI approaches to planning have been shown to be computationally infeasible when applied to such problems [9]. More recent approaches to planning [2] have dwelt on much lower-level problem solving capabilities and a complex design problem is far beyond their horizons.

The design of a behavioural layer in a subsumption architecture seems to be design by magic, by sleight of hand, by indirection; in the sense that the desired behaviour can often be described as an emergent by-product of rule-following which does not explicitly mention that behaviour [14]. Emergence is in itself nothing magic as a phenomenon, if it is considered as emergence-in-the-eyes-of-the-beholder. Something can be characterised as emergent relative to an initial given description if:

1. a system can be set up which is completely described in this initial way
2. A new description of the behaviour of the system can be made which 'is useful' or 'makes sense' to an observer, and makes use of concepts outside those originally given.

To design such emergent behaviours hence requires either (a) a computationally intractable planning problem or (b) a creative act on the part of the designer — which is to be greatly admired, though impossible to formalise. In both cases it seems likely that the limits of feasibility are currently being tested.

[1] *Animat* ... simulated animal or autonomous robot.[24]

3 Let's evolve robots instead

If, however, some objective fitness function can be derived for any given architecture, there is the possibility of automatic evolution of the architecture without explicit design. Natural evolution is the existence proof for the viability of this approach, given appropriate resources. Genetic Algorithms (GAs) [11] use ideas borrowed from evolution in order to solve problems in highly complex search spaces, and it is here suggested that GAs, suitably extended in their application, are a means of evading the problems mentioned in the previous section.

The artificial evolution approach will maintain a population of viable genotypes (chromosomes), coding for cognitive architectures, which will be inter-bred and mutated according to a selection pressure. This pressure will be controlled by a task-oriented evaluation function: the better the robot performs its task the more evolutionarily favoured is its cognitive architecture. Rather than attempting to hand design a system to perform a particular task or range of tasks well, the evolutionary approach will allow their gradual emergence.

The sleight-of-hand, or indirection, problem mentioned above, is avoided with GAs in that there is no need for any assumptions about means to achieve a particular kind of behaviour, as long as this behaviour is directly or implicitly included in the evaluation function.

Brooks' subsumption approach was mentioned above as a contrast to the dogmatic assumptions of functional decomposition implicit in much of traditional robotics. Nevertheless, it is similarly not necessary to be dogmatically committed to an exclusively behavioural decomposition. By allowing both types of decomposition, the evolutionary process will determine where in practice the balance should lie in the robots' cognitive architecture.

4 Related Work

A number of researchers have speculated on the use of evolutionary techniques for mobile robot programming [3, 23], though no practical applications have been reported. Some have shown the method to be viable for simulated robots in highly simplified simulated worlds [1], but have not had to face the exponential increase in complexity that follows from progress from toy worlds into the real world.

In December 1991 Brooks reported at ECAL-91 in Paris [8] that he was starting work on an evolutionary approach to robotics, based on Koza's Genetic Programming techniques [16]. He acknowledged the practical necessity of largely using simulations, and

stressed the dangers of using simulated worlds rather than real worlds. We would endorse two further points he makes; firstly that by evolving the control program incrementally the search space can be kept small at any time; and secondly that symmetries or repeated structures should be exploited so that only a single module needs to be evolved, which is then repeatedly used.

Where we would differ from Brooks is that we do not believe that a programming language is the right medium of control to be evolved. As a technique for evolving programs, Koza's Genetic Programming does seem to be the best framework available, with the programs effectively treated as trees which a crossover operator, swapping subtrees, handles sensibly. But any programming language which incorporates a DO-WHILE loop (or anything equivalent) suffers from the possibility — indeed with large programs the probability — of non-halting programs. There is no algorithm for detecting such programs, short of waiting forever, and the usual technique used by Koza is to define *forever* as, for instance, a few hundred program steps, rejecting those programs that have not halted by then. This inherent brittleness of programs, in that a single mutation to a 'viable' program can turn it into a non-halter, is we believe a fundamental barrier to progress in this direction as programs get larger.

The success of work by Ray [21] on evolving replicating 'organisms' and 'parasites' based on machine-code instructions has been interpreted by some as demonstrating that the inherent brittleness of partial recursive program languages can be avoided. But it is avoided in this case because there is no externally imposed evaluation which requires each machine-code program to be 'run until it halts' for its fitness to be evaluated. The real time of the 'organisms' is measured in terms of program steps. But in contrast to this, when robot control is by program, the assumption is made that when inputs are assessed at each externally dictated clock-tick, the outputs can be calculated 'instantaneously', or at least before the next clock-tick. This is usually an adequate approximation *except specifically* in the case of non-halting programs.

5 The use of Neural Networks

Because of just these issues, we advocate real-time dynamical neural networks as the medium of evolution, rather than programming languages. A feedback loop in such a network might be considered analogous to a non-halting program if it results in permanent oscillations in the network, but there is no problem of 'illegality' here if indeed such an oscillation in real time is fed through to the outputs.

Neural networks as the medium of control — but not in an evolutionary context — have been used in such projects as ALVINN with the CMU Navlab [20] for real robots. This used a backpropagation network with a human driver as the teaching input; the task was to learn to drive the Navlab van down roads using the input from a video camera. The limitations of using a synthesized environment meant that training sequences were derived from real roads, with techniques developed to ensure that examples of varied and bad driving were also available to be learnt from.

Important work on an evolutionary approach to simulated robotics using neural networks by Beer has come to our notice while this paper was in preparation [5]. He explores the evolution of continuous-time recurrent neural networks as a mechanism for adaptive agent control, using as example tasks chemotaxis, and locomotion-control for a six-legged insect-like agent. The starting point taken can be characterised by this quotation:

> ...animals are endowed with nervous systems whose dynamics are such that, when coupled with the dynamics of their bodies and environments, these animals can engage in the patterns of behavior necessary for their survival.

Thus the time constants of the model neurons in the networks is of as much significance as the weights on the connections between them; and there is no reason to restrict networks to feedforward.

The technique used by Beer and Gallagher to determine the time constants, thresholds and connection weights is a standard GA, as implemented in the GENESIS package. They report success in their objectives, and tried a number of GA variations in the use of selection only, or crossover only, or mutation only. Their discussion of the issues involved is to be recommended to anybody working in this area. They do not emphasise in the way Brooks (and Pomerleau) does the important distinctions to be made between simulations and reality, as their work reported here is confined to simulations; nevertheless a neural net approach such as theirs should be inherently much less brittle in the presence of noise than a programming approach.

For solving a particular small-scale problem the evolution of a continuous-time recurrent neural net, with due attention to the necessity for noise in simulations as discussed below, is what we would advocate. For more complex systems we believe it is necessary to analyse carefully the use of evolutionary techniques

to produce incrementally more complex architectures [13]. Standard GA practice is no longer adequate.

6 An Incremental Species approach

Though an animal should not be considered as a solution to a problem posed 4 billion years ago, in the short term adaptations in a species may be usefully interpreted as solving particular problems for that species. So when using the evolution of animals as a source of ideas for the evolution of *animats*, GAs should be used as a method for searching the space of possible adaptations to an existing *animat*, not as a search through the complete space of *animats*.

This of course has strong resemblances to Brooks' incremental approach, wherein 'low-level' behaviours are wired in and thoroughly debugged, before the next layer of behaviour is carefully designed on top of them. The difference with the approach we advocate is that of substituting evolution for design.

In [13] a framework is developed for extending GAs to deal with phenotypes of increasing complexity, and hence necessarily genotypes of increasing length. It is shown that, within this Species Adaptation Genetic Algorithm (SAGA) framework, after any initial evolutionary search a population will inevitably be largely converged, and hence thereafter be evolving as a *species*; and any individual changes in genotype length in the next generation, associated with increase in complexity, must necessarily be restricted to relatively slight changes.

Whereas in standard GAs the crossover operator is the main engine driving the search, from an initial population widely spread across the search space towards homing in on an optimal area, in SAGA the main operators operating on a converged population are mutation and change-length operators. The mutation operator, rather than being at a very low-level background rate, is of the order of magnitude of 1 mutation per genotype; a figure, incidentally, supported by results given by Beer and Gallagher [5]. The crossover operator is used to 'mix-and-match' any improvements made by the other operators, and must be carefully crafted to minimise disruption.

7 Evaluation

The human roboticist should, so we argue, relinquish the design role to the evolutionary mechanisms of an extended GA, once the capabilities being sought for the robot go beyond toy domains. But unrestricted evolution will have no reason to produce *animats* that the human wishes for, and hence for practical purposes the human must take on a role analogous to the plant breeder or cattle breeder, both of whom act so as to influence the future course of existing species. Hence the task of the human will be to design a set of robot tasks of increasing complexity (and ways to evaluate them), leading from the capabilities of existing robots towards those capabilities required in the future. This is non-trivial, although the creative human input to this process will be at a higher level of abstraction than current design at the nuts and bolts level.

Such a sequence in evolutionary time parallels the addition of layers of behaviour in a subsumption architecture designed by hand — for instance progression from obstacle-avoidance to wandering to wall-following to simple navigation. The score of an individual robot will at any one time be based on a function of the scores on each of the individual tasks, and at any time a new task can be added which contributes to the score. Species evolution within the SAGA framework limits the search process to the appropriate adaptations to networks with already established capabilities. A price that has to be accepted here is that only a minute portion of the global search space is covered; but no other method avoids this once we are beyond toy domains.

8 Morphogenesis

Approaches to the problem of evolving connectionist network architectures that have been taken include those in [17, 19]. All these have used some form of GA to search through a pre-defined finite space of possible network architectures. In other words, at a more or less sophisticated level, the basic architecture has been defined with some parameters left as variables, and the GA has been used to tweak the parameters to optimal values. Work by Harp [12] with variable length genotypes allows for networks of potentially arbitrary complexity, requiring a careful crossover operator that exchanges homologous segments.

To move to an open-ended space of possible architectures, rather than adjusting parameters in a finite space, a genetic coding able to produce a network from a genotype of indefinite length must be used. Since in this case there can be no direct mapping from each pre-defined section of the genotype to a parameter of the network, the process of translating the genotype into the network cannot all be done in parallel, and hence at least part of the translation must be a sequential process. This is in contrast to standard GA practice, where in functional terms the translation from genotype to phenotype could well be done in parallel, and the genotype is in a sense a simple description of the salient characteristics of the phenotype.

Once the genotype is interpreted sequentially, the production of the phenotype is now a developmental

process, where the sequence of interpretation is of significance. So it is necessary to find some appropriate developmental language, such that the genotype determines the developmental process, which results in a network. It should be noted that in the real world, the appropriate parts of the DNA in each cell are translated sequentially; and the development of the form of the body, including the brain, from the population of cells which constitute it, is itself a sequential process of immense and little-understood complexity. The information in the DNA is too small by many orders of magnitude to uniquely specify each connection in the brain, and hence there must be some form of modularity in this specification. One presumes that there are building blocks (for instance, layers with a certain pattern of projection for the connectivity between them) that are useful in many different parts of the brain; rather than each occurrence being individually coded for in the genotype, such a building block may be coded for in perhaps just one place, and this information used many times over in the development of the brain structure.

In biology, morphogenesis is one of the least-understood areas. In the artificial evolution of neural networks we know of no very satisfactory method for modelling a developmental process from genotype to network, and this currently presents the greatest challenge. For the time being *ad hoc* solutions must be used.

9 The need for simulation

Artificial evolution requires the evaluation of members of a sizeable population over the course of many generations. In the case of the evolution of autonomous robot control systems, it would take far too long to do all of these evaluation in the real world, and instead most must be done in simulation. It is crucial that the simulation is kept as closely in step with reality as possible. A number of techniques can be used to this end. Firstly, the simulation can be calibrated at regular intervals by carefully testing the architectures evolved in the real robot. Serious discrepancies should be ironed out. Secondly, accurate simulations of the inputs to the robot sensors and the reactions to the actuators should be based on carefully collected empirical data. Noise must be taken into account at all levels. In order to acquire the desired level of accuracy it may be necessary to use a mixed hardware/software simulation in which simulated signals are fed into hardware sensors or actuators and the response is read directly. The use of low resolution sensing makes this approach feasible.

A range of unstructured dynamic environments should be used in the simulation. A cognitive architecture that has evolved to cope with a range of such environments is much more likely to be robust than one evolved to operate in a single well structured world.

Of course it is not possible to simulate the world in all its rich detail, even the world of a robot with low resolution senses. This would be a serious problem if the traditional techniques of robot control were to be used; fragile techniques that cannot cope with even slight divergence from the details of their internal models. However, the use of adaptive noise tolerant units, such as neural nets, as the key elements of the control system means that 100% accuracy is not required. Discrepancies between the simulations and the real world, as long as they are not too big, can be treated as noise; the system can adapt to cope with this.

When it comes to evaluation of a robot architecture, since the results on individual runs will vary with noise, a number of runs should be made. Taking the minimum score achieved over a number of runs as the final score will implicitly favour robust designs.

In the long term, as the robots become more sophisticated and their worlds more dynamic, will the simulation run out of steam? The simulation of a medium resolution visual system with, for instance, motion detection pre-processing is painfully slow on today's hardware. The quantity and quality of input information required, especially in a dynamic environment, would make further heavy demands. There is no such thing as a free lunch, and it will not be surprising if this is the barrier in the long term. Nevertheless in the medium term, working with minimal sensory inputs within the capabilities of current simulations, an enormous range of behaviours, far beyond those currently hand-designed, can be generated through relatively small-scale cognitive architectures which remain amenable to an evolutionary approach.

These issues are discussed further in a paper submitted to Artificial Life 3 [15].

10 An example simulation

A real robot assembled in the Engineering Department at Sussex has been simulated using this methodology. The behaviour of the motors propelling the wheels has been modelled for the outputs, as have inputs from whisker and bumper touch sensors. Simulation of a low-resolution insect-type visual system will be added. This is part of an ongoing project at Sussex to develop an evolutionary approach to robotics, with increasingly sophisticated tasks leading to navigation using learnt landmarks.

A relatively complex floor-plan of an area in which

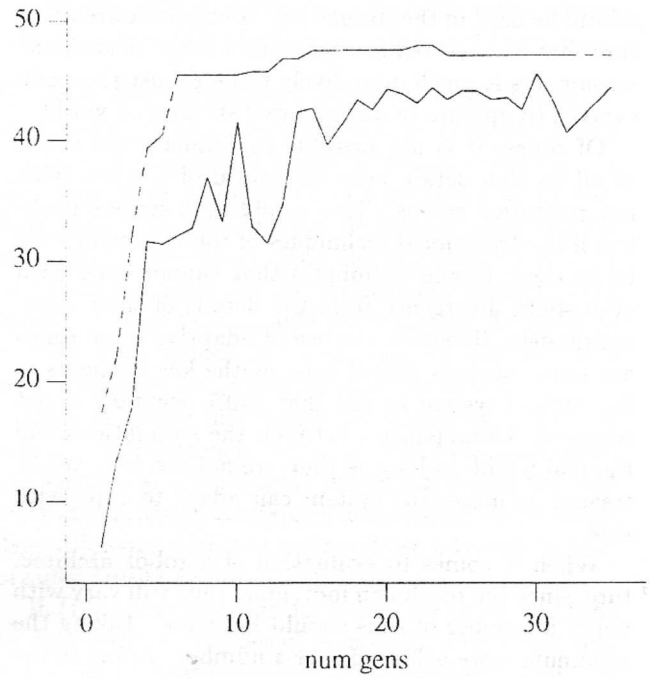

Figure 1: Results of experiment.

the robot can wander is modelled, including walls, doors and pillars. The continuous physical motion of the robot, including contact with obstacles, is accurately calculated. Realistic values of noise are added both to the inputs and to the output signals from the neural network control architecture, to the inputs to internal nodes, and also to the deflections on hitting obstacles.

The network has a fixed number of input nodes associated with the sensors, of output nodes associated with the motors, and a variable number of internal nodes. Connections, genetically determined, can be between any nodes in either direction, and can either contribute to the inputs of the target node, or act as a veto on its outputs. The sum of the inputs to a node (with noise) is passed through a sigmoid function — in fact the output is zero until the input sum reaches a minimum threshold, thereafter a linear function of the input sum until a maximum threshold is reached, whereupon the output remains fixed at its maximum value. The continuous behaviour of the network is approximated with (fine) discrete time slices. The output signals are polled after a number of such time slices; to counter distorting periodic effects this number has some variance about a mean.

The genetic coding for the architecture has to allow for a variable number of internal nodes, and a variable number of connections from each node. Interpreted sequentially along the genotype, firstly the input nodes are coded for, each preceded by a marker. For each node, the threshold values of the sigmoid can first be coded, thereafter a variable number of groups each representing a single connection from that node. Each group specifies whether it is an additive or a veto connection, and then the target node indicated by jump-type and by jump-size. The jump-type allows for relative addressing by jumps forward or backwards along the genotype ordering; or for absolute addressing relative to the start or the end of the genotype. The listing of the variable number of connections from the given node is terminated when a genetic marker indicates that the next node has been reached.

The internal nodes and output nodes are handled similarly in succession with their own identifying genetic markers; it is these markers for the internal nodes that identify the variable numbers of these. The variable length of the resulting genotypes necessitates a careful crossover operator which exchanges homologous segments. In keeping with SAGA principles, when a crossover between two parents results in an offspring of different length, such changes in length (although allowed) are restricted to a minimum.

Figure 1 shows the results for an experiment in which a control network was evolved for a robot with four whiskers and front and back bumpers. The evaluation function was designed to encourage wandering. Robust control networks were favoured by scoring each genotype several times for a single evaluation. On each scoring run the robot faced a different set of situations because of the noise in the system. The minimum of the scores achieved was taken as the evaluation figure. The robots were started from rest with no initialising signals. Internal noise was sufficient to allow fitter nets to settle into useful initial states. The bottom line on the graph shows the evaluation figure for the best individual in each generation. The top line shows the best score achieved by any member of the population for any of the runs making up its evaluation set. The fact that these two lines converge indicates that more and more robust networks appeared. The theoretical maximum score (which could only be achieved with a perfect map and navigation system) was 51.5. Clearly, very good control networks have been evolved in only a few generations. Useful robust behaviour has been achieved without internal representations or any kind of predefined strategy. Figure 2 shows a short run by a robot controlled by one of these networks, the circles are pillars and the lines walls. The robot's whiskers are shown moving through objects as a matter of convenience only. It can be seen that the net-

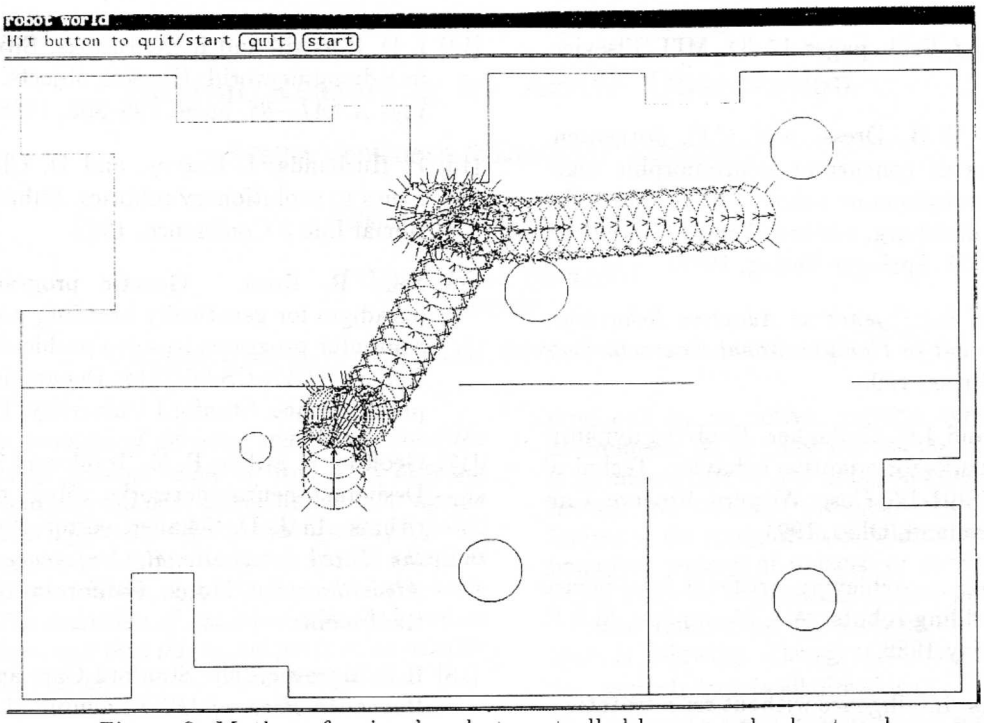

Figure 2: Motion of a simple robot controlled by an evolved network.

work generates a 'move in straight line at full speed behaviour' in free space and various rotational movements to avoid obstacles and move away from walls. Members of earlier generations had far more random behaviours and spent most of their time colliding with objects or sitting still.

11 Conclusions

The type of cognitive architecture suitable for robust control of a robot may well not be easily decomposable into significant large modules; either by function or by behaviour. Yet humans only seem to be good at design when such decompositions can be effected. Automation of such a design process by turning it into a planning problem is computationally intractable. Natural evolution is the existence proof for an alternative approach, which is here advocated.

Reasons have been presented for thinking that programs, as the description of the control system, are not a suitable medium for evolution. Recurrent neural networks operating in real time have been proposed as a suitable medium, in that they avoid the brittleness of programs and are tolerant to noise. In evolving architectures of increasing complexity, variable length genotypes become essential, and standard genetic algorithms need to be extended to deal with this. In the consequent species evolution the pattern of search is very different from normal GA search, and the role of genetic operators is changed. The morphogenesis of a network from a genotype has been discussed; while there are *ad hoc* solutions, a completely satisfactory solution is yet to be found.

The potential conflict between simulations and reality has been discussed, and in particular the important role of noise. Results have been reported from an ongoing project to develop robot cognitive architectures using this evolutionary methodology.

Acknowledgements

We'd like to thank Harry Barrow, Dave Cliff, Tony Simpson, Jean-Arcady Meyer, Alexandre Wallyn and Simon Goss for helpful discussions about this work or related issues.

References

[1] D. H. Ackley and M. L. Littman. Interactions between learning and evolution. In C. G. Langton, J. D. Farmer, S. Rasmussen, and C. Taylor, editors, *Artificial Life II: Proceedings Volume of Santa Fe Conference Feb. 1990.* Addison Wesley: volume XI in the series of the Santa Fe Institute Studies in the Sciences of Complexity, 1991.

[2] P.E. Agre and D. Chapman. What are plans for? In P. Maes, editor, *Designing Autonomous Agents: Theory and Practice from Biology to En-*

gineering and Back, pages 17–34. MIT/Elsevier, 1990.

[3] J. Barhen, W.B. Dress, and C.C. Jorgensen. Applications of concurrent neuromorphic algorithms for autonomous robots. In R. Eckmiller and C.v.d. Malsburg, editors, *Neural Computers*, pages 321–333. Springer-Verlag, 1987.

[4] R. D. Beer. *Intelligence as Adaptive Behaviour: An Experiment in Computational Neuroethology.* Academic Press, 1990.

[5] R.D. Beer and J.C. Gallagher. Evolving dynamic neural networks for adaptive behavior. Technical Report CES-91-17, Case Western Reserve University, Cleveland, Ohio, 1991.

[6] R. A. Brooks. Achieving artificial intelligence through building robots. A.I. Memo 899, M.I.T. A.I. Lab, May 1986.

[7] R.A. Brooks. Intelligence without representation. *Artificial Intelligence*, 47:139–159, 1991.

[8] Rodney A. Brooks. Artificial life and real robots. In *Proceedings of the First European Conference on Artificial Life.* MIT Press/Bradford Books, Cambridge, MA, 1992.

[9] David Chapman. Planning for conjuctive goals. *Artificial Intelligence*, 32(3):333–377, 1987.

[10] D. T. Cliff. Computational neuroethology: A provisional manifesto. In J.-A. Meyer and S.W. Wilson, editors, *From Animals to Animats: Proceedings of The First International Conference on Simulation of Adaptive Behavior*, pages 29–39. MIT Press/Bradford Books, Cambridge, MA, 1991.

[11] David E. Goldberg. *Genetic Algorithms in Search, Optimization and Machine Learning.* Addison-Wesley, Reading, Massachusetts, USA, 1989.

[12] S.A. Harp and T. Samad. Genetic synthesis of neural network architecture. In L. Davis, editor, *Handbook of Genetic Algorithms*, pages 202–221. Van Nostrand Reinhold, 1991.

[13] Inman Harvey. Species adaptation genetic algorithms: The basis for a continuing SAGA. In *Proceedings of the First European Conference on Artificial Life.* MIT Press/Bradford Books, Cambridge, MA, 1992.

[14] I. D. Horswill and R. A. Brooks. Situated vision in a dynamic world: Chasing objects. In *Proceedings AAAI—88*, pages 796–800, 1988.

[15] P. Husbands, I. Harvey, and D. Cliff. Central issues in evolutionary robotics. Submitted to Artificial Life 3 Conference, 1992.

[16] John R. Koza. Genetic programming: A paradigm for genetically breeding populations of computer programs to solve problems. Technical Report STAN-CS-90-1314, Department of Computer Science, Stanford University, 1990.

[17] Geoffrey F. Miller, P. M. Todd, and S. U. Hegde. Designing neural networks using genetic algorithms. In J. D. Schaffer, editor, *Proceedings of the Third International Conference on Genetic Algorithms*, San Mateo, California, 1989. Morgan Kaufmann.

[18] H.P. Moravec. The Stanford Cart and the CMU Rover. In *Proc. of IEEE*, volume 71, pages 872–884, 1983.

[19] H. Muhlenbein and J. Kindermann. The dynamics of evolution and learning - towards genetic neural networks. In R. Pfeifer, Z. Schreter, F. Fogelman-Soulie, and L. Steels, editors, *Connectionism in Perspective*, pages 173–197. Elsevier Science Publishers B.V. (North-Holland), 1989.

[20] D. A. Pomerleau. Efficient training of artificial neural networks for autonomous navigation. *Neural Computation*, 3:88–97, 1991.

[21] Thomas S. Ray. An approach to the synthesis of life. In J.D. Farmer, C.G. Langton, S. Rasmussen, and C. Taylor, editors, *Artificial Life II.* Addison-Wesley, 1992.

[22] S.J. Rosenschein. Formal theories of knowledge in AI and robotics. *New Generation Computing*, 3:345–357, 1985.

[23] P. Viola. Mobile robot evolution. Bachelors thesis, M.I.T., 1988.

[24] S.W. Wilson. Knowledge growth in an artificial animal. In J. Grefenstette, editor, *Proceedings of the First International Conference on Genetic Algorithms and their applications.* Lawrence Erlbaum Assoc., 1985.