THE ARTIFICIAL EVOLUTION OF CONTROL SYSTEMS

P Husbands, I Harvey, D Cliff, A Thompson, N Jakobi

University of Sussex, England

ABSTRACT

Recently there have been a number of proposals for the use of artificial evolution as a radically new approach to the development of robot control systems. This paper explains and justifies the evolutionary robotics methodology developed and used at Sussex University. Results are presented from research into the concurrent evolution of dynamical network controllers and visual sensor morphologies for an autonomous mobile robot. Successful experiments to evolve robot controllers directly in hardware, thus allowing advantage to be taken of intrinsic silicon physics, are also discussed. The implications of recent work in using high quality engineering simulations to evolve control systems for real robots will are covered.

INTRODUCTION

When designing a control system for a robot, there are at least three major problems:
• It is not clear *how* a robot control system should be decomposed.
• Interactions between separate sub-systems are not limited to directly visible connecting links, but also include interactions mediated *via the environment.*
• As system complexity grows, the number of potential interactions between sub-parts of the system grows *exponentially.*

Classical approaches to robotics have often assumed a primary decomposition into Perception, Planning and Action modules. Many people now see this as a basic error [3, 2]. Brooks acknowledges the latter two problems above in his subsumption architecture approach. This advocates slow and careful building up of a robot control system layer by layer, in an approach that is explicitly claimed to be inspired by natural evolution — though each new layer of behaviour is wired in by hand design.

An obvious alternative approach is to abandon hand design and explicitly use evolutionary techniques to incrementally evolve increasingly complex robot control systems. Unanticipated interactions between sub-systems need not directly bother an evolutionary process where the only benchmark is the behaviour of the whole system. The basic notion of Evolutionary Robotics, then, is as follows. The evolutionary process, based on a genetic algorithm [8], involves evaluating, over many generations, whole populations of control systems specified by artificial genotypes. These are interbred using a Darwinian scheme in which the fittest individuals are most likely to produce offspring. Fitness is measured in terms of how good a robot's behaviour is according to some evaluation criterion. The work reported here forms part of a long-term study to explore the viability of such an approach in developing interesting adaptive behaviours in visually guided autonomous robots, and, through analysis, in better understanding general mechanisms underlying the generation of such behaviours. It is one of the strands of the research program of the Evolutionary and Adaptive Systems Group, School of Cognitive and Computing Sciences, University of Sussex. For further details see e.g. [5].

WHAT AND HOW TO EVOLVE

We must choose appropriate building blocks for evolution to work with. Primitives manipulated by the evolutionary process should be at the lowest level possible. Any high level semantic groupings inevitably incorporate the human designer's prejudices. We agree with Brooks [3] in dismissing the classical Perception, Planning, Action decomposition of robot control systems. Instead we see the robot as a whole — body, sensors, motors and 'nervous system' — as a dynamical system coupled (via sensors and motors) with a dynamic environment [1]. Hence the genotype should encode at the level of the primitives of a dynamical system.

One such system is a dynamic recurrent neural net (DRNN), with genetic specification of connections and of the timescales of internal feedback. These DRNNs can in principle simulate the temporal behaviour of any finite dynamical system, and are equivalent (with trivial transformations) to Brooks' subsumption architectures. We also deliberately introduce internal noise at the nodes of DRNNs, with two effects. First, it makes possible new types of feedback dynamics, such as self-bootstrapping feedback loops and oscillator loops. Second, it helps to make more smooth the fitness landscape on which the GA is operating.

In ER a genotype will specify the control system of a robot which is expected to produce appropriate behaviours. The number of components required may be unknown *a priori*; and when using incremental evolution, through successively more difficult tasks, the number of components needed will increase over time. Such incremental evolution calls for *GAs as adaptive improvers* rather than *GAs as optimisers*.

Species Adaptation Genetic Algorithms (SAGA) were developed for this purpose [7]. It was shown that progress through such a genotype space of increasing complexity will only be feasible through gradual increases in genotype length; this implies the evolution of a *species* — the population is largely genetically converged. With successive generations, selection is a force which tends to move such a population up hills on a fitness landscape, and keep it centred around a local optimum; whereas mutation explores outwards from the current population. For a given selection pressure, there is a maximum rate of mutation which simultaneously allows the population to retain a hold on its current hill-top, whilst maximising search along relatively high ridges in the landscape, potentially towards higher peaks. In SAGA, this means that rank-based selection should be used to maintain a constant selective pressure, and mutation rates should be of the order of 1 mutation per genotype [7].

SIMULATION?

A number of New-Wave roboticists have consistently warned of the dangers of working with over-simple unvalidated robot simulations [3, 2, 12]. Indeed, as Smithers has pointed out [12], the word simulation has been somewhat debased in the fields of AI, robotics, and animat research. Many so-called simulations are abstract computer models of imaginary robot-like entities, not carefully constructed models of *real* robots. Whereas these abstract models can be very useful in exploring some aspects of the problem of control in autonomous agents, great care must be taken in using them to draw conclusions about behaviour in the real world. Unless their limitations are recognised, they can lead to both the study of problems that do not exist in the real world, and the ignoring of problems that do [2]. Behaviours developed in a simulation worthy of the name must correspond closely to those achieved when the control system is down-loaded onto the real robot.

One area of New-Wave robotics where these issues may be particularly pertinent is evolutionary robotics [9]. Two of the earliest papers on this topic both stressed the likelyhood of having to work largely in simulation to overcome the time consuming nature of doing all the evaluations in the real world [4, 9]. However, both discussed the potential problems with simulations and remarked on the great care that would have to be taken. In [4], Brooks was highly sceptical[1]:

> There is a real danger (in fact, a near certainty) that programs which work well on simulated robots will completely fail on real robots because of the differences in real world sensing and actuation – it is very hard to simulate the actual dynamics of the real world.

and later,

> ...[sensors] ...simply do not return clean accurate readings. At best they deliver a fuzzy approximation to what they are apparently measuring, and often they return something completely different.

But since the aim of evolutionary robotics is to produce working real robots, if simulations are to be used, these problems must be faced. The question is how. In [9] it is argued that:

- The simulation should be based on large quantities of carefully collected empirical data, and should be regularly validated.
- Appropriately profiled noise should be taken into account at all levels.
- The use of networks of adaptive noise tolerant units as the key elements of the control systems will help to 'soak up' discrepancies between the simulation and the real world.
- Noise added *in addition to* the empirically determined stochastic properties of the robot may help to cope with the inevitable deficiencies of the simulation by blurring them. A control system robust enough to cope with such an envelope-of-noise may handle the transfer from simulation to reality better than one that cannot deal with uncertainty over and above that inherent in the underlying simulation model.

With these questions in mind Jakobi [10] built a simulator, *Khepsim*, and conducted a number of experiments. The simulation is based on a spatially continuous, two dimensional model of the underlying real world physics and not on a look-up table approach as in [11] (parameters were set using empirical information from a *Khepera*). This affords greater generality with respect to new environments and unmodelled situations although at some computational expense. The simulation is updated once every 100 simulated milliseconds: the rate at which the inputs and outputs of the neural network control architectures are processed. This results in relatively coarse time slicing, some of the effects of which may be moderated by noise.

Neural networks evolved in simulation evoked qualitatively similar behaviour on the real robot, the

[1] It is likely that these comments were influenced by experiences with devices rather different to the robot used in the experiments described later. This issue is returned to in the Conclusions.

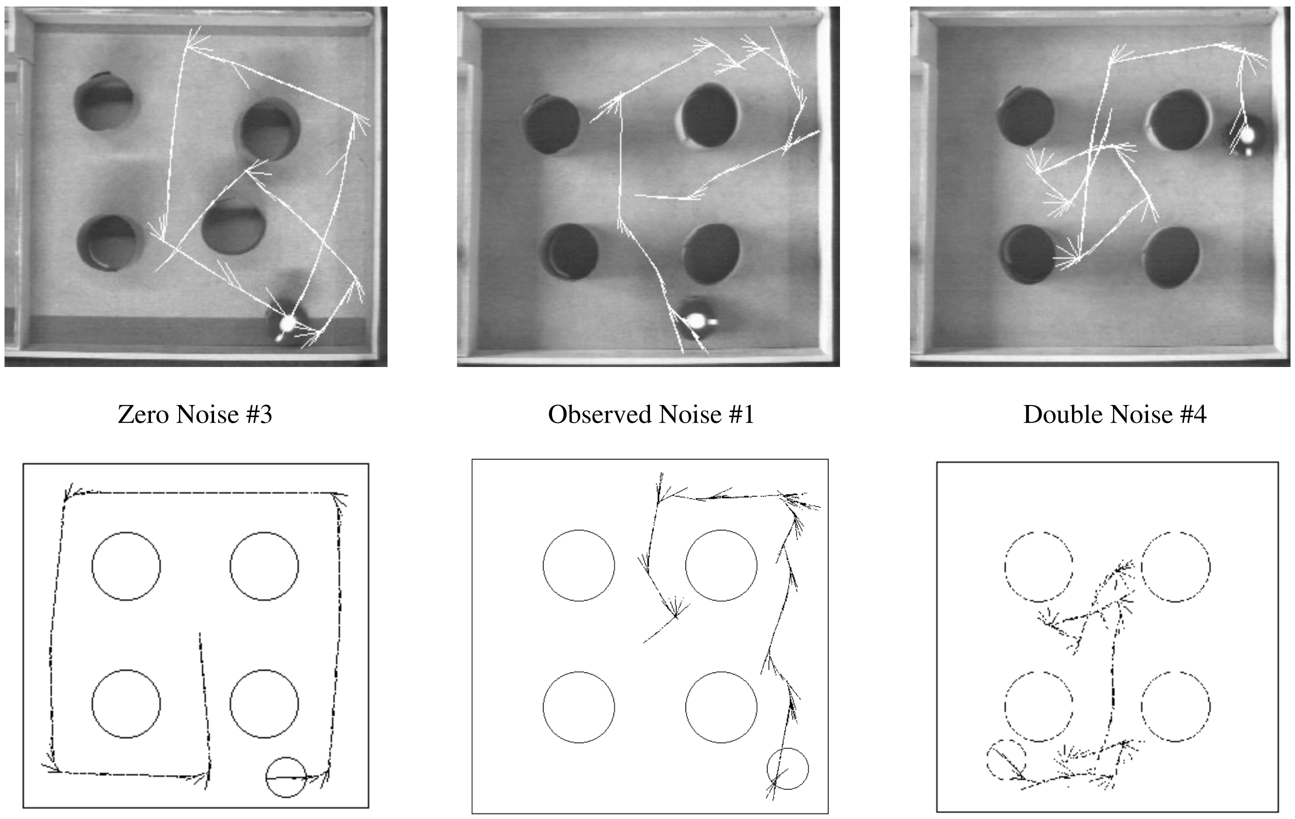

Figure 1: Obstacle avoidance: from simulation to reality. These six pictures display the situated and simulated behaviours of three different neural network controllers, one taken from each noise class. The #s refer to Table 1.

correspondence being a matter of degree rather than binary valued. The following experiments were designed to inspect two factors that affect this correspondence: the nature of the behaviour itself and the level of noise present in the simulation.

For each of two behaviours, obstacle avoiding and light seeking, three sets of five evolutionary runs were performed, one set for each of three different noise levels. These three noise levels were set at zero noise, observed noise and double observed noise. Observed noise (on sensors, motors etc.) refers to a roughly Gaussian distribution with standard deviation equal to that empirically derived from experiments. Double observed noise refers to the same distribution with double the standard deviation. As expected, it was found that in general, networks evolved in an environment that is less noisy than the real world will behave more noisily when downloaded onto the Khepera and, conversely, networks evolved in an environment that is noisier than the real world will behave less noisily when downloaded. Simulation to situation correspondence seems to be maximised when the noise levels of the simulation have similar amplitudes to those observed in reality. The behaviours shown in Figures 1 and 2 graphically illustrate this.

For both obstacle avoidance and light-seeking, the set of experiments running under observed noise obtained the highest average behaviour score. In a zero noise environment, brittle 'hit or miss' strategies tend to evolve which either score incredibly well or incredibly badly on each fitness trial, depending on their initial random starting positions. Although noise, in general, blurs the fitness landscape, reducing the possibility of 'hit or miss' strategies evolving (since they are far more likely to 'miss' rather than 'hit'), too much randomness in the environment, as in the double noise case, ensures that the same genotypes may again achieve very different scores on two otherwise identical fitness evaluations. A balance between these two cases seems to be achieved at the observed noise level.

One conclusion from these experiments is that simulations can *in some circumstances* be good enough to be used for artificial evolution, with the resulting designs successfully downloaded onto a real *Khepera*. It seems likely that there are strong limitations on how far it is realistic to extend this approach.

A second conclusion is that the noise used in such simulations should be at a level similar to that observed experimentally. If there is a significant difference in noise levels, then whole different classes of behaviours become available which, while acquiring high fitness scores in simulation, necessarily fail to work in reality. This is true both for too little noise, and for too much noise.

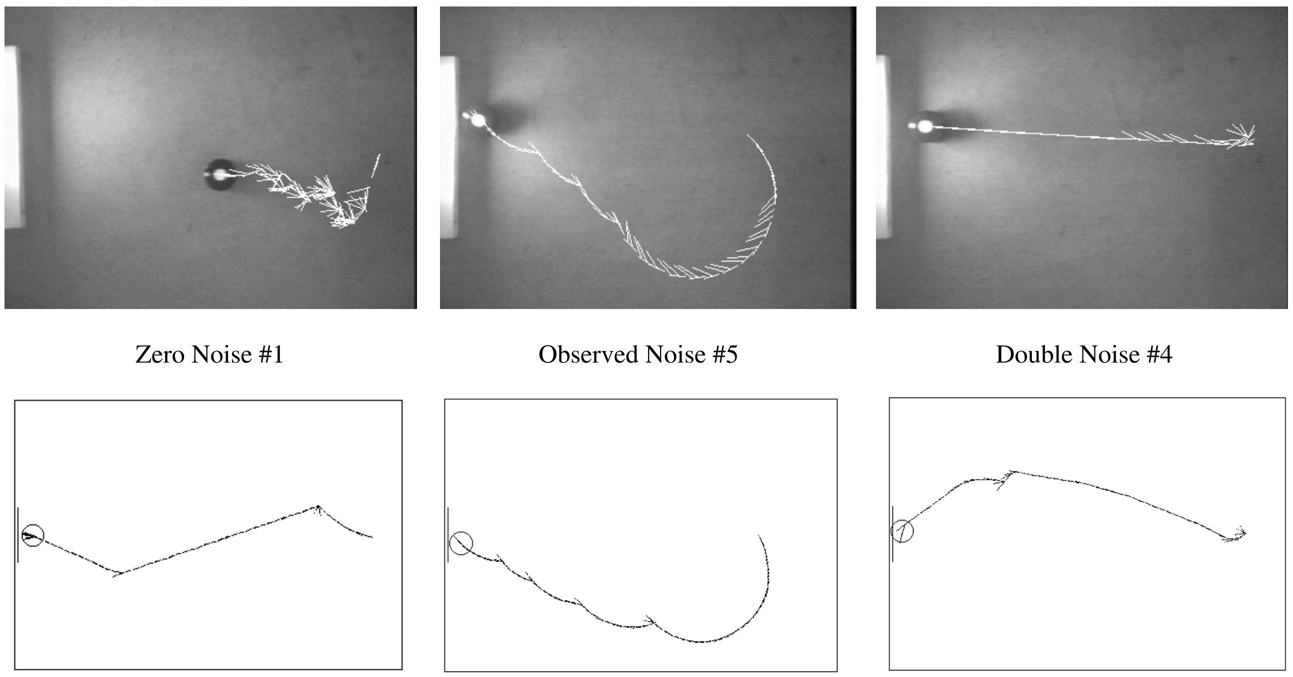

Figure 2: Light seeking: from simulation to reality. These six pictures display the situated and simulated behaviours of three different neural network controllers, one taken from each noise class. The #s refer to Table 2.

REAL WORLD EVOLUTION

The research described in the previous section showed that control systems evolved in carefully constructed simulations, with an appropriate treatment of noise, transfer extremely well to reality, generating almost identical behaviours in the real robot. However, this example involved relatively simple robot-environment interaction dynamics. Once even low-bandwidth vision is used, simulations become altogether more problematic. They become difficult and time consuming to construct and computationally very intensive to run. Hence evolving visually guided robots in the real world becomes a more attractive option. The experiment described next uses a piece of robotic equipment specially designed to allow the real-world evolution of visually guided behaviours — the Sussex gantry-robot.

THE GANTRY-ROBOT

The gantry-robot is shown in Figures 3 and 4. The robot is cylindrical, some 150mm in diameter. It is suspended from the gantry-frame with stepper motors that allow translational movement in the X and Y directions, relative to a co-ordinate frame fixed to the gantry . The maximum X (and Y) speed is about 200mm/s. Such movements, together with appropriate rotation of the sensory apparatus, correspond to those which would be produced by left and right wheels. The visual sensory apparatus consists of a CCD camera pointing down at a mirror inclined at 45^o to the vertical. The mirror can be rotated about a vertical axis so that its orientation always corresponds to the direction the 'robot' is facing. The visual inputs undergo some transformations en route to the control system, described later. The hardware is designed so that these transformations are done completely externally to the processing of the control system.

The control system for the robot is run off-board on a fast personal computer, the 'Brain PC'. This computer receives any changes in visual input by interrupts from a second dedicated 'Vision PC'. A third (single-board) computer, the SBC, sends interrupts to the Brain PC signalling tactile inputs resulting from the robot bumping into walls or physical obstacles. The only outputs of the control system are motor signals. These values are sent, via interrupts, to the SBC, which generates the appropriate stepper motor movements on the gantry.

The Brain PC runs the top-level genetic algorithm and during an individual evaluation, it is dedicated to running a genetically specified control system for a fixed period. At intervals during an evaluation, a signal is sent from the Brain PC to the SBC requesting the current position and orientation of the robot. These are used in keeping score according to the current fitness function. The Brain PC receives signals, to be fed into the control system, representing sensory inputs from the Vision PC and the SBC. The visual signals are derived from averaging over genetically specified circular receptive patches in the camera's field of view.

This setup, with off-board computing and avoidance of tangled umbilicals, means that the apparatus

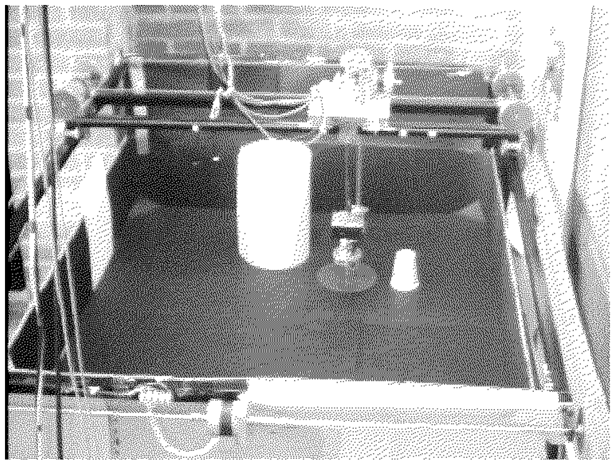
Figure 3: *The Gantry viewed from above. The horizontal girder moves along the side rails, and the robot is suspended from a platform which moves along this girder.*

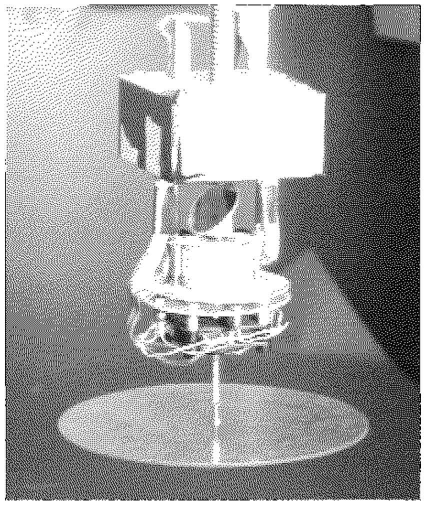
Figure 4: *The gantry-robot. The camera inside the top box points down at the inclined mirror, which can be turned by the stepper-motor beneath. The lower plastic disk is suspended from a joystick, to detect collisions with obstacles.*

can be run continuously for long periods of time – making artificial evolution feasible. A top-level program automatically evaluates, in turn, each member of a population of control systems. A new population is produced by selective interbreeding and the cycle repeats. For full technical details of the system see [6].

EXPERIMENTAL SETUP

Full details of the experimental setup for the gantry-robot can be found in [6]. This paper also explains in full the genetic encodings used and the control system primitives manipulated by the GA. Experiments conducted with the gantry-robot to date have all involved relatively simple vision based navigation tasks. The experiment described below was one of a series where a converged population of robots was evolved through a series of increasingly complex behaviours.

These were based around the evolution of control architectures built from recurrent dynamic realtime networks, where the primitives were the nodes in a network, and links between them. There were no restrictions on network topologies, arbitrarily recurrent nets being allowed. When some of these nodes are connected to sensors, and some to actuators, the network acts as a control system, generating behaviours in the robot.

Rather than imposing a fixed visual sampling morphology, we believe a more powerful approach is to allow the visual morphology to evolve along with the rest of the control system. Hence we genetically specify regions of the robot's visual field to be sub-sampled, these provide the only visual inputs to the control network. It would be desirable to have many aspects of the robot's morphology under genetic control, although this is not yet technically feasible.

Starting from a converged population of robots that could move forward, but little else, the first task was to move to a large white target from random starting points and orientations. Once this was being achieved, the task was changed to approaching a small white target and evolution continued.

RECTANGLES AND TRIANGLES

The experiment then continued with a distinguish-between-two-targets task. Two white paper targets were fixed to one of the gantry walls; one was a rectangle, the other was an isosceles triangle with the same base width and height as the rectangle. The robot was started at four positions and orientations near the opposite wall such that it was not biased towards either of the two targets. The evaluation function $\mathcal{E}_3$, to be maximised, was:

$$\mathcal{E}_3 = \sum_{i=1}^{i=20} [\beta(D_{1_i} - d_{1_i}) - \sigma(D_{2_i}, d_{2_i})] \qquad (1)$$

where D_1 is the distance of target-1 (in this case the triangle) from the gantry origin; d_1 is the distance of the robot from target-1; and D_2 and d_2 are the corresponding distances for target-2 (in this case the rectangle). These are sampled at regular intervals, as before. The value of β is $(D_1 - d_1)$ unless d_1 is less than some threshold, in which case it is $3 \times (D_1 - d_1)$. The value of σ (a penalty function) is zero unless d_2 is less than the same threshold, in which case it is $I - (D_2 - d_2)$, where I is the distance between the targets; I is more than double the threshold distance. High fitnesses are achieved for approaching the triangle but ignoring the rectangle. It was hoped that this experiment might demonstrate the efficacy of concurrently evolving the visual sampling morphology along with the control networks.

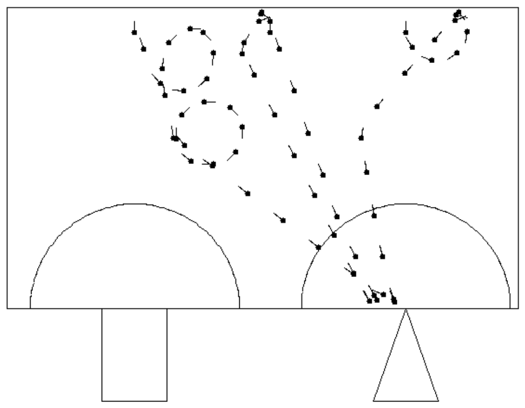

Figure 5: *Behaviour of a fit individual in the two target environment. The rectangle and triangle indicate the positions of the targets. The semi circles mark the 'penalty' (near rectangle) and 'bonus score' (near triangle) zones associated with the fitness function. In these 4 runs the robot was started directly facing each of the two target, and twice from a position midway between the two targets; once facing into the wall and once facing out.*

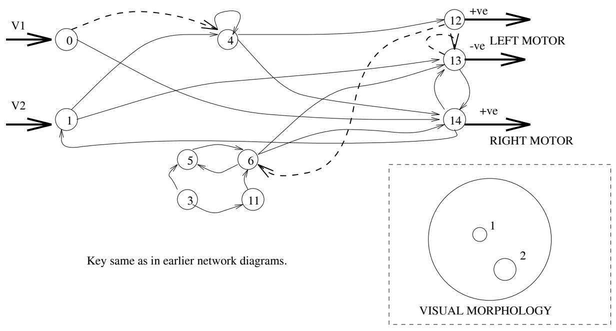

Figure 6: *Active part of the control system that generated fit behaviour for the rectangle and triangle experiment. Visual morphology shown inset.*

After about 15 generations of a run using as an initial population the last generation of the incremental small target experiment, fit individuals emerged capable of approaching the triangle, but not the rectangle, from each of the four widely spaced starting positions and orientations. The behaviour generated by the fittest of these control systems is shown in Figure 5. When started from many different positions and orientations near the far wall, and with the targets in different positions relative to each other, this controller repeatedly exhibited very similar behaviours to those shown.

The active part of the evolved network that generated this behaviour is shown in Figure 6. The evolved visual morphology for this control system is shown inset. Only receptive fields 1 and 2 were used by the controller.

Detailed analyses of this evolved system can be found in [6]. To crudely summarise, unless there is a *difference* in the visual inputs for receptive fields 1 and 2, the robot makes rotational movements. When there is a difference it moves in a straight line. The visual sensor layout and network dynamics have evolved such that it fixates on the sloping edge of the triangle and moves towards it.

TRANSIENT BEHAVIOUR

Time plots of behaviour against this difference in visual inputs consistently revealed an interesting non-reactive feature to the robot's behaviour. Figure 7 shows such a plot. The behaviour axis (Z) is discretized into simple observable motor behaviours such as straight line motion, rotating on the spot, movement in the arc of a circle and so on. The final part of the plot, a line parallel to the time axis and terminating at the point marked 'finish' at the right hand side of the cube, represents the straight line motion when the robot has fixated on the triangle edge and is moving towards it. The parallel line above this and immediately to the left represents a short lived transient behaviour which such plots revealed *always* occurred when the visual signal difference become large. Briefly, the onset of a large difference triggers a short sharp rotational movement which has very different consequences depending on whether the robot has fixated on a vertical or sloping edge. With a vertical edge, the rotation tends to move both receptors off the target, the visual signals become very different and rotational behaviours ensue. However, with a sloping edge, the rotation is not enough to move both receptive fields off the target; the visual signal difference is still there and a straight line motion follows. This is illustrated in Figure 8. This behaviour can be interpreted as a kind of 'checking' of edge orientation.

EVOLVABLE HARDWARE

The work discussed so far has generally used a genetically specified dynamical system as the control system for a simulated or real robot. But this dynamical system, for instance when conceptualised as a DRNN, has in practice been implemented on a computer. There is a related approach of evolving control systems directly onto hardware, which has been taken within our group by Thompson [13].

This work is *intrinsic* hardware evolution, in that for each genetically specified piece of hardware, the actual hardware is tested *in situ*; as contrasted with extrinsic hardware evolution, where simulations of the hardware are evaluated during evolution. The actual low-level physics of the hardware can be utilised, and the realtime dynamics operate at their proper timescales. Our notion of the nature of electronic

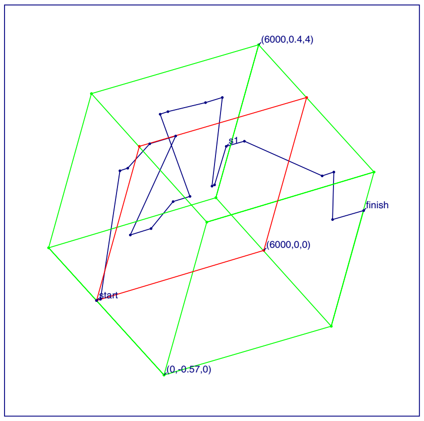

Figure 7: *Time plot of behaviour against difference in visual inputs for receptive fields 1 and 2. The time axis (X) runs left to right, the visual signal difference axis (Y) runs bottom to top on the lower face of the cube, the behaviour axis (Z) runs from lower to top face of the cube. See text for further details.*

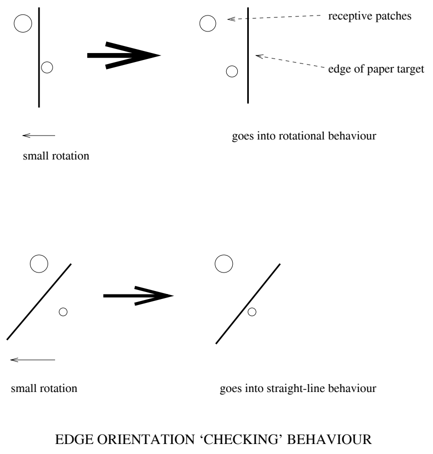
receptive patches
edge of paper target
small rotation
goes into rotational behaviour
small rotation
goes into straight-line behaviour
EDGE ORIENTATION 'CHECKING' BEHAVIOUR

Figure 8: *The top part of the figure illustrates the outcome of the transient 'checking' behaviour when the receptive fields straddle a vertical edge, and the bottom part shows the same when they straddle a sloping edge.*

systems is heavily biased by our design methodologies and the constraints applied to facilitate their abstractions, so evolvable hardware demands a radical rethink of what electronic circuits can be. Both the spatial structure (modularity) and the temporal structure (synchronisation and the rôle of phase in general) need to be considered.

With digital design by conventional methods, care must be taken to prevent switching transients (a feature absent from the designer's model) from affecting the system's overall behaviour. Usually, this means that the circuit is broken into modules, the internal transient dynamics of which are hidden from each other. Real physical electronic circuits are continuous-time dynamical systems. They can display a broad range of dynamical behaviour, of which discrete-time systems, digital systems and even computational systems are but subsets. These subsets are much more amenable to design techniques than dynamical electronic systems in general, because the restrictions to the dynamics that each subset brings support design abstractions. Intrinsic EHW does not require abstract models, so there is no need to constrain artificially the dynamics of the reconfigurable hardware being used.

In particular, there no longer needs to be an *enforced* method of controlling the phase (temporal coordination) in reconfigurable hardware originally intended to implement digital designs. The phase of the system does not have to be advanced in lock-step by a global clock, nor even the local phase-controlling mechanisms of asynchronous digital design methodologies imposed.

In one simulation experiment [13], Thompson demonstrated that a network of high-speed logic gates could be evolved to oscillate at a much slower timescale. At the start of the experiment, each of 100 logic nodes was assigned a real-valued propagation delay, selected uniformly randomly from the range 1.0 to 5.0 nanoseconds. The genotype specified the boolean function performed at each node, and the connectivity between nodes. Evolution was performed on a population of such genotypes, which were evaluated on the basis of the average period between logic transitions at one specified node: the closer to a square wave oscillation of 1kHz, the fitter. After 40 generations under this selection pressure, the output of the best individual was approximately $4\frac{1}{2}$ thousand times slower than the best of the random initial population, and was six orders of magnitude slower than the propagation delays of the nodes. Fitness was still rising at generation 40 when the simulation was terminated.

Thompson then used artificial evolution to design a real EHW circuit as an on-board controller for a two-wheeled autonomous mobile robot (diameter of 46cm, a height of 63cm) required to display simple wall-avoidance behaviour in an empty $2.9\text{m}\times 4.2\text{m}$ rectangular arena. For this scenario, the d.c. motors driving the wheels were not allowed to run in reverse and the robot's only sensors were a pair of time-of-flight sonars rigidly mounted on the robot, pointing left and right. The sonars fire simultaneously five times a second; when a sonar fires, its output changes from logic **0** to logic **1** and stays there until the first echo is sensed at its transducer, at which time its output returns to **0**.

Conventional design methods would preprocess the sonar output pulses to give indications of the range to the nearest objects. From these, a central controller would be a hardware implementation of a finite-state machine (FSM), with the next-state and output functions designed so as to compute appropriate motor speeds for each wheel. For each wheel, a pulse-width modulator would take the binary representation of motor speed from the central controller

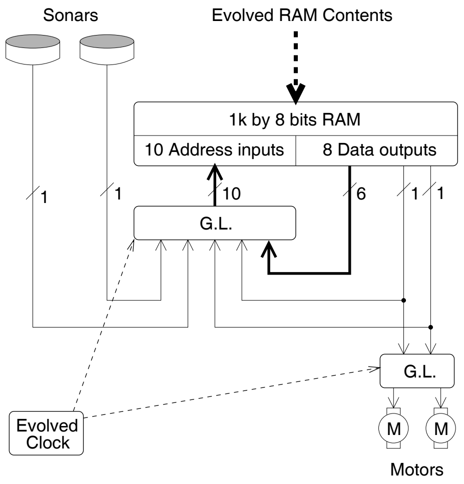

Figure 9: The hardware implementation of the evolvable DSM. 'G.L.' stands for a bank of genetic latches: it is under genetic control whether each signal is passed straight through asynchronously, or whether it is latched according to the global clock of evolved frequency.

and vary the mark:space ratio of pulses sent to the motor accordingly.

It would be possible to evolve the central controller FSM as intrinsic EHW by implementing the next-state and output functions as look-up tables held in an off-the-shelf random access memory (RAM) chip.[2]. The FSM would then be specified by the bits held in the RAM, which could be reconfigured under the control of each individual's genotype in turn. There would be no benefit in evolving this architecture as hardware, however, because the electronics is constrained to behave in accordance with the FSM design abstraction: all of the signals are synchronised to a global clock to give clean, deterministic state-transition behaviour as predicted by the model. Consequently, the hardware would behave identically to a software implementation of the same FSM.

The alternative approach taken here is to relax the constraint of synchronisation of all signals, and place it under evolutionary control. The global clock frequency is under genetic control, as also is the choice of whether each signal is synchronised (latched) by the clock or asynchronous. This new architecture is termed a *Dynamic State Machine* (DSM). It is not a finite-state machine because a description of its state must include the temporal relationship between the asynchronous signals, which is an real-valued analogue quantity. In the conventionally designed control system there was a clear sensory/control/motor decomposition (timers/controller/pulse-width-modulators), communicating in atemporal binary representations which hid the real-time dynamics of the sensorimotor systems, and the environment linking them, from the central controller. Now, the evolving DSM is intimately coupled to the real-time dynamics of its sensorimotor environment, so that real-valued time can play an important rôle throughout the system. The evolving DSM can explore special-purpose tight sensorimotor couplings because the temporal signals can quickly flow through the system being influenced by, and in turn perturbing, the DSM on their way.

[2]This is the well known 'Direct Addressed ROM' implementation of an FSM

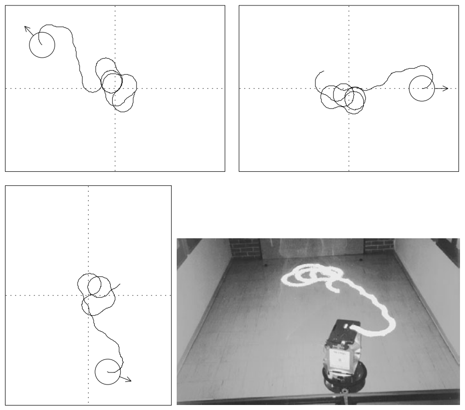

Figure 10: Wall avoidance in virtual reality and (bottom right) in the real world, after 35 generations. The top pictures are of 90 seconds of behaviour, the bottom ones of 60.

For convenience, evolution took place with the robot in a kind of 'virtual reality.' The real evolving hardware controlled the real motors, but the wheels were just spinning in the air. The wheels' angular velocities were measured, and used by a real time simulation of the motor characteristics and robot dynamics to calculate how the robot would move. The sonar echo signals were then artificially synthesised and supplied in real time to the hardware DSM. Realistic levels of noise were included in the sensor and motor models, both of which were constructed by fitting curves to experimental measurements, including a probabilistic model for specular sonar reflections.

Fig. 10 shows the excellent performance which was attained after 35 generations, with a good transfer from the virtual environment to the real world. The robot is drawn to scale at its starting position, with its initial heading indicated by the arrow; thereafter only the trajectory of the centre of the robot is drawn. The bottom-right picture is a photograph of behaviour in the real world, taken by double-exposing a picture of the robot at its starting position, with a long exposure of a light fixed on top of the robot, moving in the darkened arena. If started repeatedly from the same position in the real world, the robot follows a different trajectory each time (occasionally *very* different), because of real-world noise. The robot displays the same qualitative range of behaviours in the virtual world, and the bottom pictures of Fig. 10 were deliberately chosen to illustrate this. One of the evolved wall-avoiding DSMs was analysed, and found to be going from sonar echo signals to motor pulses using only 32 bits of RAM and 3 flip-flops (excluding clock generation): highly efficient use of hardware resources, made possible by the absence of design constraints.

Further experiments are demonstrating that evolution may be an effective method of producing hardware tolerant to single-stuck-at (SSA) faults in the RAM's memory array. Evolutionary search with a population, tends to seek high local areas in the fitness landscape, rather than single high points. A local, area is here defined as nearby points in genotype space, points reachable from each other by a single mutation, or a very small number. Experiments reported in [14] show, by simulating the effects of adverse SSA faults, that fault-tolerance does indeed develop under evolution.

CONCLUSIONS

Evolutionary Robotics is a research area in its infancy; the tests for all newborn AI philosophies are whether they can grow up into the real world, and scale up with increasing complexity. In the evolutionary experiments at Sussex we have started to demonstrate the possibilities in simulation, on real robots, and directly in silicon.

References

[1] R.D. Beer and J.C. Gallagher. Evolving dynamic neural networks for adaptive behavior. *Adaptive Behavior*, 1(1):91–122, 1992.

[2] R.A. Brooks. Intelligence without reason. In *Proceedings IJCAI-91*, pages 569–595. Morgan Kaufmann, 1991.

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

[4] Rodney A. Brooks. Artificial life and real robots. In F. J. Varela and P. Bourgine, editors, *Proceedings of the First European Conference on Artificial Life*, pages 3–10. MIT Press/Bradford Books, Cambridge, MA, 1992.

[5] D. Cliff, I. Harvey, and P. Husbands. Explorations in evolutionary robotics. *Adaptive Behavior*, 2(1):73–110, 1993.

[6] I. Harvey, P. Husbands, and D. Cliff. Seeing the light: Artificial evolution, real vision. In D. Cliff, P. Husbands, J.-A. Meyer, and S. Wilson, editors, *From Animals to Animats 3, Proc. of 3rd Intl. Conf. on Simulation of Adaptive Behavior, SAB'94*, pages 392–401. MIT Press/Bradford Books, 1994.

[7] Inman Harvey. Evolutionary robotics and SAGA: the case for hill crawling and tournament selection. In C. Langton, editor, *Artificial Life III*, pages 299–326. Santa Fe Institute Studies in the Sciences of Complexity, Proceedings Vol. XVI, Addison-Wesley, Redwood City CA, 1994.

[8] John Holland. *Adaptation in Natural and Artificial Systems*. University of Michigan Press, Ann Arbor, USA, 1975.

[9] P. Husbands and I. Harvey. Evolution versus design: Controlling autonomous robots. In *Integrating Perception, Planning and Action, Proceedings of 3rd Annual Conference on Artificial Intelligence, Simulation and Planning*, pages 139–146. IEEE Press, 1992.

[10] N. Jakobi, P. Husbands, and I. Harvey. Noise and the reality gap: The use of simulation in evolutionary robotics. In F. Moran, A. Moreno, J.J. Merelo, and P. Chacon, editors, *Advances in Artificial Life: Proc. 3rd European Conference on Artificial Life*, pages 704–720. Springer-Verlag, Lecture Notes in Artificial Intelligence 929, 1995.

[11] S. Nolfi, D. Floreano, O. Miglino, and F. Mondada. How to evolve autonomous robots: Different approaches in evolutionary robotics. In R. Brooks and P. Maes, editors, *Artificial Life IV*, pages 190–197. MIT Press/Bradford Books, 1994.

[12] Tim Smithers. On why better robots make it harder. In D. Cliff, P. Husbands, J.-A. Meyer, and S. Wilson, editors, *From Animals to Animats 3, Proc. of 3rd Intl. Conf. on Simulation of Adaptive Behavior, SAB'94*, pages 54–72. MIT Press/Bradford Books, 1994.

[13] A. Thompson. Evolving electronic robot controllers that exploit hardware resources. In F. Moran, A. Moreno, J.J. Merelo, and P. Chacon, editors, *Advances in Artificial Life: Proc. 3rd European Conference on Artificial Life*, pages 640–656. Springer-Verlag, Lecture Notes in Artificial Intelligence 929, 1995.

[14] A. Thompson, I. Harvey, and P. Husbands. Unconstrained evolution and hard consequences. In E. Sanchez and M. Tomassini, editors, *Towards Evolvable Hardware*. Springer-Verlag, 1996.