Explorations in Evolutionary Robotics

Dave Cliff*
University of Sussex

Inman Harvey†
University of Sussex

Phil Husbands‡
University of Sussex

We discuss the methodological foundations for our work on the development of cognitive architectures, *or control systems, for situated autonomous agents. Our focus is the problems of developing sensorimotor control systems for mobile robots, but we also discuss the applicability of our approach to the study of biological systems. We argue that, for agents required to exhibit sophisticated interactions with their environments, complex sensorimotor processing is necessary, and the design, by hand, of control systems capable of such processing is likely to become prohibitively difficult as complexity increases. We propose an automatic design process involving artificial evolution, wherein the basic building blocks used for evolving cognitive architectures are noise-tolerant dynamical neural networks. These networks may be recurrent and should operate in real time. The evolution should be incremental, using an extended and modified version of a genetic algorithm.*

Practical constraints suggest that initial architecture evaluations should be done largely in simulation. To support our claims and proposals, we summarize results from some preliminary simulation experiments in which visually guided robots are evolved to operate in simple environments. Significantly, our results demonstrate that robust visually guided control systems evolve from evaluation functions that do not explicitly require monitoring visual input. We outline the difficulties involved in continuing with simulations and conclude by describing specialized visuorobotic equipment, designed to eliminate the need for simulated sensors and actuators.

Key Words: evolutionary robotics; autonomous agents; genetic algorithms; SAGA; sensorimotor coordination; neural networks

* School of Cognitive and Computing Sciences and Neuroscience IRC, School of Biological Sciences, University of Sussex, Brighton BN1 9QH, UK; davec@cogs.susx.ac.uk

† School of Cognitive and Computing Sciences, University of Sussex, Brighton BN1 9QH, UK; inmanh@cogs.susx.ac.uk

‡ School of Cognitive and Computing Sciences, University of Sussex, Brighton BN1 9QH, UK; philh@cogs.susx.ac.uk

1 Introduction

There is a similarity between the engineering problem of creating a successful autonomous robot intended to act in complex noisy environments and the scientific problem of proposing a plausible model for mechanisms underlying the generation of adaptive behavior in an animal. Both tasks essentially involve *designing* a system that satisfies given constraints, and this design process requires skilled creative insight, something we believe is impossible to formalize. The work presented here is motivated by the concern that for both problems, the sheer complexity of designing cognitive systems may severely curtail the possibilities for progress if we continue to rely on purely intuitive and manual techniques. For this reason, we are attempting to develop automatic techniques that will be of use both in building intelligent autonomous robots and in understanding how animals are "built." The bulk of this article concentrates on the development of such a technique for building control systems, or *cognitive architectures*, for autonomous robots, but we do discuss briefly how this might be applied to problems in biology.

In our attempts to develop artificial cognitive systems, we assume that the study of biological cognitive systems can be highly informative. The recent increase of interest in artificial neural network research has given rise to a proliferation of work in the literature that lays claims to biological inspiration or biological plausibility. Yet the vast majority of such research is motivated almost entirely by physiological observations (i.e., that the nervous systems underlying cognition in animals can, at certain levels of abstraction, be modeled as networks of simple processing units operating in parallel). We, in common with a number of other authors, believe that it is necessary to study cognition from a much wider biological perspective.

In particular, we consider it essential to acknowledge that in the context of cognition, neural networks in animals serve the function of coordinating perception and action in order to generate adaptive behaviors, a point long stressed by Arbib (see, e.g., Arbib, 1989). In the biology literature, study of the neural mechanisms underlying the generation of behaviors is known as *neuroethology* (*ethology* being the study of animal behavior), and so we consider our work to be closer to computational neuroethology (Beer, 1990; Cliff, 1991b) than to computational neuroscience (Sejnowski, Koch, & Churchland, 1988). However, adopting an ethological perspective also requires that ecological factors be taken into account: Whether a particular behavior is adaptive depends crucially on the immediate environment and ecological niche of the animal.

The study of neural function in ethological, ecological, and evolutionary contexts is standard practice in biology but much less common in artificial neural network research. A major portion of this article is devoted to a discussion of how adoption

of this wider biological framework applies to the development of artificial cognitive systems.

One distinguishing consequence of this focus on artificial neural networks as generators of behaviors within particular environments is a dependence on working with *animats* (i.e., artificial creatures [Wilson, 1985; Brooks, 1986, 1990b]), either real physical robots or simulated autonomous agents situated in some virtual reality. It should be noted that the problems encountered in ensuring adaptive sensorimotor coordination in real robots, or in virtual agents within appropriate simulated worlds, are sufficiently similar to the problems encountered by animals that there is a strong potential for fruitful interdisciplinary study. For example, McFarland (1990) has proposed a number of mathematical analogs between adaptive behaviors in animals and adaptive behaviors in robots.

The work described here involves using a form of simulated evolution, based on an extended genetic algorithm, to develop neural network control systems for mobile robots. Taking a further cue from biology, our definition of *control system* includes parameters determining the robot's sensors (and, in principle, motors), so the sensorimotor *morphology* of the robot is evolved *concurrently* with the controller networks.

As hinted at earlier, our rationale for employing artificial evolution centers on a belief that the task of designing sensorimotor control systems in which there are many interactions between subcomponents is sufficiently difficult that as required behavioral and, hence, neural complexity increases, purely manual techniques are unlikely to be sufficient. Automatic techniques have been successfully developed in other fields, such as operations research, to aid in complex tasks previously performed entirely by hand. Although the need for automation in other fields lends some support to our claim that it will be needed in the field of autonomous agent design, this does not guarantee that techniques developed for other domains will transfer easily to ours.

Indeed, in order to exploit fully the potential of simulated evolution, it is necessary to employ a form of genetic algorithm that is significantly extended from the straightforward genetic algorithms used in applications that can be reduced to function optimization (Goldberg, 1989; Holland, 1975). Necessary adaptations to genetic algorithms are summarized here and discussed fully in Harvey (1992a, 1992b, 1993).

To begin, we present our rationale, which proposes that an evolutionary approach to the design of robots can be expected to supercede design by hand, and we explore issues arising from this. After setting the stage with these theoretical considerations, we report on preliminary simulation experiments involving the use of an extended genetic algorithm to evolve control networks for simple robots equipped with a few

touch sensors and extremely low-resolution visual perception. The simulations are not naive; they are based on observations of a real robot and attempt to model the physics of its interactions with the world. Following this, we offer some notes on those topics that we consider, on the basis of our experiences to date, to be the most important areas for future research, chief among which is the need to transfer from simulations to real robots.

2 Rationale

2.1 Interesting robots need distal sensing

There is a growing body of literature describing the successful construction of autonomous mobile robots built according to principles of *behavioral decomposition* (where the control architecture is composed of a heterarchical arrangement of modules, each of which is responsible for generating particular behaviors [Brooks, 1985]) and some evidence that similar organizational principles may be found in the nervous systems of animals (Altman & Kien, 1989). One fundamental behavioral competence required of any animat is the ability to move and navigate in cluttered, dynamical, and uncertain surroundings.

In autonomous robot navigation, a variety of sensor technologies commonly are used to facilitate navigation. The development of a successful autonomous robot depends on finding a satisfactory combination of exteroceptors and interoceptors.[1] Truly autonomous robots will require interoceptors to monitor significant internal states such as power level and degree of wear or damage. Although interoception is an important subject, we will limit our discussion here to the topic of exteroception for navigation.

The majority of research projects in behavioral robotics have not employed vision as a primary exteroceptive mechanism (notable exceptions are Connell, 1990, and Horswill, 1993). Most commonly, mechanosensory (tactile) "whiskers" and "bumpers" or active ranging devices (such as ultrasonic depth sensors) have been employed. Such sensors often are used as essentially *proximal* sensing devices: that is, they provide reliable data only for the immediate surroundings of the robot. Robots with only proximal sensors are limited in their navigation strategies. A particularly common strategy is wall following, where the robot must always maintain sensory contact with a sizable static external surface, such as a wall of the robotics laboratory.

1 In the biology literature, an *exteroceptor* is a receptor (i.e., sensor) that detects stimuli external to an animal (e.g., light or sound), whereas an *interoceptor* senses or detects stimuli that arise inside the animal (e.g., blood pressure).

In some restricted behavioral or ecological niches, wall following is a satisfactory navigation strategy, yet there are a number of behaviors that are highly impracticable to expect of robots equipped solely with proximal exteroceptors. Clearly, more sophisticated navigation competences will require *distal* sensing. Of the technologies available for distal sensing, vision is an attractive option given the large body of literature devoted to the study of biological visual systems and the construction of computer vision systems. Furthermore, the passive nature of visual sensing offers energy economies that are attractive in a robot that carries its power supplies on-board.

Vision provides a rich source of information concerning an agent's external environment. We are aware that for many behavioral niches in many environments, a variety of sensory modalities may be needed, and that in certain cases, visible light may be an inappropriate sensing medium. Nevertheless, because visual sensing has the advantages mentioned earlier, our initial work has concentrated on vision combined with tactile sensing. We believe that our concerns about the difficulty of design (see later) are equally applicable to other modalities.

In designing or evolving a visual system, a number of factors must be taken into account. Significant issues include:

- The discretization of the sampling of the optic array (i.e., how many pixels do we want in the images our robot samples, and what sort of geometry should the image have [a square raster being not necessarily ideal]?). The number of pixels in the image largely determines the bandwidth of the visual processing channels.
- The angular extent of the visual system's field of view (should the robot be equipped with 360-degree vision, or will a more restricted field of view suffice?).
- The visual angular resolution of the robot's optics (should the visual system employ a uniform resolution or have some sort of spatially variant "foveal" (nonuniform) vision system?). Many animals have resolution that varies across the visual field. Typically, this is a result of the need for high-resolution vision for certain tasks (identifying prey or mates) coupled with a need for a wide field of view, sampled at a lower resolution. There are many niches for robots in which spatially variant vision can be used to good effect.

That many animals, particularly insects, successfully occupy their ecological niches using comparatively low-resolution vision as a primary means of distal exteroception indicates that such an approach (as opposed to high-resolution, high-bandwidth vision) is worth exploring, in the first instance at least. However, even low-resolution systems are difficult to design.

2.2 Interesting robots may be too difficult to design

Traditional approaches to the development of vision-based autonomous robot control systems have made only modest progress, with fragile and computationally very expensive methods. This is due largely to the traditional implicit assumption of functional decomposition—the assumption that perception, planning, and action can be analyzed and synthesized independently of one another.

In contrast, recent work at the Massachusetts Institute of Technology bases robot control architectures around behavioral decomposition (Brooks, 1986, 1991). In theory, this involves analyzing independent behaviors of an animat such that each behavior can be wired in all the way from sensor input to motor output. Simple behaviors are wired in first, and then more complex behaviors are added as separate layers, affecting earlier layers only by means of suppression or inhibition mechanisms.

In designing such systems, it is extremely difficult to foresee all possible interactions with the environment and between separate parts of the robot itself (Brooks, 1991; Moravec, 1983). Such interactions can be both internal and via the environment. Designing appropriate cognitive architectures is a task with inherently explosive complexity, which is likely to scale much faster than the number of layers or modules within the architecture: It can scale with the number of possible interactions between modules. Therefore, it seems entirely feasible that the successes of early behavior-based architectures, designed by hand, may be more difficult to reproduce as the sophistication of the desired behaviors increases beyond the relatively primitive competences demonstrated thus far: The complexity of manual design of behaviorally decomposed architectures may become so difficult that automation of the design process is necessary. Therefore, though manual design of such architectures should not be ruled out entirely, we are studying methods by which the control systems of autonomous agents can be automatically generated from a specification of the behaviors to be exhibited by the agent.

2.3 Let's evolve robots instead

We have ruled out the use of traditional artificial intelligence techniques, such as symbolic reasoning or planning methods, on the grounds of computational intractability (Chapman, 1987). Moreover we have, for the time being at least, withheld from attempting to employ some form of self-organizing system. To rely wholly on self-organization seems implicitly to accept a *tabula rasa* view of the organization of cognitive structures. As far as we are aware, there is very little evidence in the animal kingdom for neonate animals being *wholly* devoid of behavior-generating cognitive mechanisms and indulging in spontaneous self-organization of their nervous systems shortly after birth; rather, basic *genetically specified* behavioral competences are present. It is our aim to emulate this process, to employ a form of artificial evolution that

encodes for control architectures which give rise to desired behaviors in the robots. Nevertheless, the majority of advanced animals do undergo a period of ontogenetic development, which can be strongly influenced by environmental factors. As this work advances, we will allow evolution to control parameters governing self-organization and adaptation within the lifetime of an individual robot, but initially we will work with hard-wired controllers.

If some objective fitness measure can be determined 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 use ideas borrowed from evolution in order to solve problems in highly complex search spaces, and it is here suggested that genetic algorithms, suitably extended, are a means of evading the design problems mentioned in section 2.2. A number of researchers have speculated on the use of evolutionary techniques for mobile robot programming (Barhen, Dress, & Jorgensen, 1987; Viola, 1988) but without any practical applications.

The artificial evolution approach requires maintaining a population of viable genotypes (chromosomes), coding for cognitive architectures, that reproduce differentially according to some selection pressure and are interbred and mutated. The selection pressure is controlled by a task-oriented evaluation function: The better the robot performs its task, the more offspring it has. Rather than attempting to hand-design a system to perform a particular set of tasks well, the evolutionary approach will allow the gradual emergence of such system capabilities.

There is no need for any assumptions about how to achieve a particular kind of behavior, as long as this behavior is directly or implicitly included in the evaluation function. Brooks's behavioral approach (Brooks, 1985) was mentioned earlier as a contrast to the dogmatic assumptions of functional decomposition implicit in much of traditional robotics. It is similarly not necessary to be dogmatically committed to an exclusively behavioral decomposition. By allowing either type of decomposition, the evolutionary process will determine whether in practice either one or neither should characterize the robot's cognitive architecture.

2.4 An incremental, open-ended approach

An animal should not be considered a solution to a problem posed 4 billion years ago. Nevertheless, in the short term, adaptations in a species may be usefully interpreted as solving particular problems for that species. Hence, when using the evolution of animals as a source of ideas for the evolution of animats, genetic algorithms should be employed as a method for searching the space of possible adaptations of an existing animat, not as a search through the complete space of animats: Successive adaptations over a long time scale can lead to long-term increases in complexity. For

this reason, although most genetic algorithms operate on fixed-length genotypes, we believe it is necessary to work instead with variable-length genotypes. Using a fixed-length genotype implicitly defines a priori a fixed-dimensionality search space. Using a variable-length genotype implies that the dimensionality of the search space is not predefined so, in principle, it should be possible to commence genetic search in a comparatively low-dimensional space (e.g., start with few genes) and allow the dimensionality to increase (e.g., allow more genes) if search in higher-dimensional spaces leads to greater fitness.

The basis for extending standard genetic algorithms to cope with this has been elucidated by Harvey (1992a, 1992b, 1993), who describes the *species adaptation genetic algorithm* (SAGA). In SAGA, the population being evolved is always a fairly genetically converged species, and increases in genotype length (or other metric of expressive power), associated with increases in complexity, can happen only very gradually.[2]

This, of course, bears a strong resemblance to Brooks's (1985) incremental approach, wherein so-called low-level behaviors are wired in and thoroughly debugged before the next layer of behavior is carefully designed on top of them. Furthermore, there are other evolutionary schemes that allow for variable-dimensionality search spaces, of which probably the best-known is Koza's form of evolutionary programming (Koza, 1990, 1992). The differences between these two approaches are discussed in the next section.

2.5 What should we evolve?

Thus far we have not fully addressed the question of why we are evolving controllers based on neural network processing architectures. Other authors have proposed evolving controllers in the form of explicit programs in some high-level language (Koza, 1992; Brooks, 1992). Here we express our views on the relative merits of the two approaches and offer discussion of what types of network are appropriate.

2.5.1 High-level programs Brooks (1992) proposes using an extension of Koza's genetic programming techniques (Koza, 1990, 1992) as the method for evolving a physical or simulated robot. One potential problem with evolving programs in a particular language is that, if the language supports partial recursion, programs to be evaluated may never halt, unless some arbitrary time-out is imposed. Brooks's behavior language (Brooks, 1990a) does not use partial recursion and hence can be evolved without this problem. Subject to the qualification that in genetic programming genotype length changes should be restricted to small steps, his approach, at first sight, seems reasonable.

2 What counts here as a small step is dependent on the ruggedness of the particular fitness landscape in genotype space. It is associated with the correlation length of such a landscape (Kauffman, 1989).

Nevertheless, we advocate that the primitives manipulated in the evolutionary process should be at the lowest level possible, which is in contrast to Brooks's use of higher-level languages (Brooks, 1992). A program written in behavior language (BL) is, in effect, a blueprint for a network of augmented finite state automata (AFSA)—the augmentation includes the addition of time delays—and the language Brooks proposes for genetic programming is an even higher-level language, GEN, which can be compiled into BL. Such a program is functionally equivalent to the recurrent real-time dynamical networks we advocate later in this article.

Whereas the search space defined by GEN and such networks may be equivalent to one another, the way in which evolution would travel through such search spaces is very different. Any high-level semantic groupings, as found in GEN, necessarily restrict the possibilities available to the evolutionary process insofar as they favor particular paths at the expense of others, compared to letting the lowest-level primitives be manipulated by genetic operators. The human designer's prejudices are incorporated within his or her choice of high-level semantics, and these restrictions give rise to a much more coarse-grained fitness landscape, with steeper precipices. It might be thought that the use of low-level primitives necessitates very many generations of evolution with vast populations before any interesting high-level behavior emerges, but our simulations show that this is not necessarily the case.

A further problem with high-level languages is that the injection of noise into anything other than the lowest levels becomes difficult to justify. For a network considered to be modeled at a physical level, it is easier to justify the insertion of noise at many points within the system and, as will be seen, this appears to have valuable effects, not least in making the fitness landscape more blurred and less rugged and hence easier for evolution.

2.5.2 Neural networks Evolutionary approaches to designing connectionist network architectures are manifold (e.g., Ackley & Littman, 1992; Beer & Gallagher, 1991; de Garis, 1990; Harp, Samad, & Guha, 1989; Kerszberg & Bergman, 1988; Miller, Todd, & Hegde, 1989; Muhlenbein & Kindermann, 1989). All have used some form of genetic algorithm to search through a predefined 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 genetic algorithm has been used to tweak the parameters to optimal values. It is argued in Harvey (1992b) that for the equivalent of robot evolution, it will be necessary to extend this to open-ended evolution, with significant implications. Nevertheless, the evolvability of connectionist networks in general has been clearly established.

It might be argued that in practice, connectionist networks are simulated on a serial computer and, in turn, that a serial Turing machine can be simulated with

a connectionist network. This does not mean that their evolvability is the same. To build a connectionist network as a virtual machine on top of a conventionally programmed computer does not alter the fact that the virtual machine may be suitable for evolutionary development whereas the underlying real machine is not: The mutations of structure are at the virtual machine level only. The price paid for this potential search efficiency, however, is the computational inefficiency of simulating one type of machine with another.

2.5.3 What sort of network? The arguments presented thus far in this article have not committed us to any particular class of neural network, and indeed our proposals are largely neutral in this respect. Nevertheless, given that adaptive behaviors are coordinated patterns of activity in space *and* time, we consider it important that networks be capable (in principle) of rich intrinsic dynamical behavior. By this we mean that the processing units ("neurons") used should be endowed with some temporal properties, even if only at the minimal level where each unit imposes a real-time delay between change in input and change in output.

Many connectionist approaches to control systems, such as feed-forward networks using back-propagation, run the risk of importing dangerous intellectual baggage from computer science (i.e., treating sensorimotor controllers as explicitly computational systems, involved solely in performing some input-output mapping). This snapshot view of cognition has been the main paradigm in artificial intelligence for much of its history. We support a much broader view, which treats autonomous agents as dynamical systems that may be perturbed by their interactions with the environment, which is also a dynamical system. This view is expressed in Maturana and Varela (1987), Beer (1992), and van Gelder (1992), and will not be developed further here. As we demonstrate in an earlier report (Husbands, Harvey, & Cliff, 1993b), neural networks can operate as dynamical systems which are *not* explicitly performing input-output mappings in the standard computational sense.

Where a dynamical real-time connectionist network differs from conventional networks is in the explicit introduction of *time*. In conventional networks, the events at the nodes or neurons happen either instantaneously and synchronously or in a randomly chosen order. The events at nodes in such networks can thus be ordered in a sequence, but there is no natural time scale on which they operate. This is not adequate for real-time behavior. The imposition of a particular time scale can be done in many ways, the simplest being to introduce time delays either within a node or in the connections between nodes. Such delays can be genetically specified. A whole new universe of possible dynamical behaviors is opened up by this extension. As noted previously, the augmentations in Brooks's AFSAs involve time delays introduced in his behavior language, for similar reasons (Brooks, 1990a).

Although we do not exclude the possibility of useful behaviors being generated by feed-forward network architectures, our current interests are in networks with recurrent connections, where there is a possibility of oscillatory and periodic activity and hence coordination of complex activity in the time domain. The practical problems of timing should be recognized. Autonomous robots will need to ensure that subcomponents, and the interactions between them, operate at appropriate time scales. This should not cause any undue problems for the implementation of evolved neural networks. As long as they operate using discrete time intervals, then even complex recurrent networks can be handled in a straightforward manner. The more general and possibly more powerful class of asynchronous continuous time networks are slightly more difficult to implement but create no significant problems.

2.5.4 Evolutionary advantages of such networks Dynamic recurrent real-time networks form an extremely general class of control systems. There are potentially far more free parameters that could be genetically determined than in synchronously stepped networks. For instance, changes in the time constants of units or neurons can radically transform the classes of behavior produced.

For a human designer to operate with some hope of success, a more constrained framework is required. Indeed, this is why higher-level languages, such as BL and GEN (mentioned earlier), are so useful to the human programmer. In constrast, the evolutionary process can manage without such constraints. Although evolution could well result in some form of modularity or hierarchy in a network's organization, the nature and organization of such modules need not be tightly restricted by human designers' prejudices.

2.6 The use of simulation

Artificial evolution requires that the members of a sizable population must be evaluated over the course of many generations. In the case of the evolution of autonomous robot control systems, to date it has been assumed it would take far too long to do all the necessary evaluations in the real world (Husbands & Harvey, 1992; Brooks, 1992; PRANCE, 1991; Barhen et al., 1987). Instead, it has been suggested that most evaluations be done in simulation. In the short to medium term, this seems a sensible strategy, but we have strong doubts about its long-term viability.

Assuming the use of simulation, it is crucial that it be kept as close to reality as possible. A number of techniques can be used to this end. First, the simulation can be calibrated at regular intervals by carefully testing the architectures evolved in the real robot. Serious discrepancies should be ironed out. Second, accurate simulations of the inputs to the robot sensors and the reactions of the actuators should be based on carefully collected empirical data from a real robot. Third, and above all, noise

must be taken into account at all levels. 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.

It is important to remember that it is not our world that is being simulated but the robot's. If the robot is intended to behave robustly over a particular class of environments, a range of appropriate environments should be used in the simulation.

If adaptive noise-tolerant units, such as neural nets, are used as the key elements of the control system, then 100 percent accuracy may not be 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.

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 preprocessing is computationally expensive. The problem here is in ensuring that the simulated visual systems correspond in a useful manner to the physical visual systems with which the robot will be equipped. Such simulations are certainly possible in principle, using computer graphics techniques and specialized hardware, but when working with real vision hardware (i.e., optical elements such as lenses, and photoelectric transducers such as CCDs), the time-costs of obtaining visual inputs are, for practical purposes, zero.

If using real visual hardware, techniques to test many generations of control systems in real worlds will have to be developed. There is no universal visual system that will cope with all environmental niches: For an evolutionary approach to be successful, methods of varying the details of the visual sampling and the subsequent processing of the visual signal are required. We are currently pursuing the development of one such technique (see section 5 for further details).

3 Experimental Methods

The following section gives details of preliminary experiments conducted to test our proposals, the results from which are presented in section 4. Most of these details are being considerably revised, extended, or replaced as our work progresses.

3.1 The robot

A real robot assembled in the engineering department at the University of Sussex has been simulated in order to test the methodology established in section 2. A plan view of the robot used in the simulations is shown in Figure 1. The robot is cylindrical in shape with two independent drive wheels toward the front and a trailing rear caster.

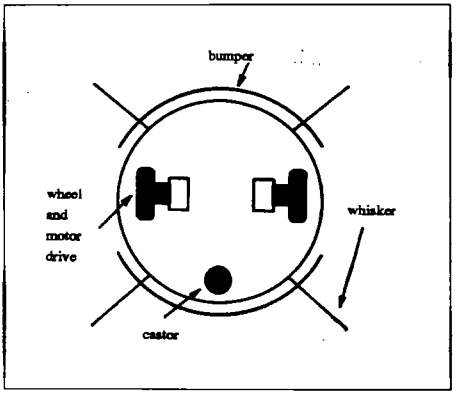


Figure 1
Plan view of simple six-sensor robot.

The signals to the drive motors for each wheel can be represented as a real value in the range $[-1.0, 1.0]$. This range is divided up into five more or less equal segments. Depending on which segment the signal falls into, the wheel will either remain stationary or rotate forward or backward at full or half speed. If the inputs to the two motors are equal in magnitude but opposite in sign, the robot will spin on the spot.

The robot is equipped with six one-bit tactile sensors: four radial whiskers arranged symmetrically about the robot's vertical axis plus front and back bumper bars. The aim of our initial experiments was to evolve neural networks that control the robot in a variety of environments. Results from earlier experiments, discussed in Harvey, Husbands, and Cliff (1993), demonstrated that our methods could be used to evolve robots that could engage in primitive navigation patterns based solely on tactile sensing in officelike environments. The primary goal in the experiments we describe was to evolve robots that could use visual perception to exhibit behaviors which would be impossible, or extremely difficult, if proximal sensing alone were employed.

For these experiments, we used a very simple world, a closed cylindrical arena with black walls and white floor and ceiling. Although this environment is comparable to those used for testing certain real visually guided robots (e.g., Franceschini, Pichon, & Blanes, 1992), it is nowhere near as visually complex as a typical real-world environment. The computational costs of providing appropriately accurate simulation data scale very poorly as the complexity of the environment increases. We deliberately chose to use such simple worlds because the costs of simulation could be contained and subsequent analysis was made easier: The prime objective was to demonstrate that our proposals are plausible, not that we could make a simulated robot wander around an empty cylindrical room.

3.2 Adding vision

Rather than imposing on the robot some visual sensors with fixed properties, it seemed much more sensible, and was in keeping with our incremental evolutionary approach, to investigate the *concurrent* evolution of visual sensors and control networks. In essence, we have started with simple very low resolution devices coupled to small networks and will work toward higher-resolution devices made useful by more complex networks generating more sophisticated behaviors. Major factors affecting how this occurs are under evolutionary control.

Because the simulated robot is based on an actual physical robot, the visual processing capabilities available to evolution must be constrained sufficiently that the designs evolved are (at least in principle) capable of being built using available hardware. In essence, this means opting for very low visual bandwidth. The total number of pixels has to be at least two or three orders of magnitude lower than that used in conventional computational vision research.[3]

For creatures that have very few photoreceptor units, the photoreceptors often have large angles of acceptance[4] and are distributed around the body so as to sample a wide visual field. These simple photoreceptor units are best thought of not as pixels in an image (or tiles in a retinal mosaic) but as simple local brightness detectors. For example, if the portion of the optic array directly above an animal suddenly goes dark while the rest of the optic array remains constant, it seems likely that something is about to drop on the animal from above, and rapid evasive action is probably an adaptive behavior in such situations. Of course, the animal does not have to construct any internal representations or reason about the cause of the darkness; it just has to do something useful.

For this reason, we have worked thus far with ultra-low-resolution vision, close in spirit to Braitenberg's *vehicles* (Braitenberg, 1984). The simulated robot has been given a few photoreceptor units, which could realistically be added to the physical robot using discrete components (e.g., photodiodes or phototransistors) with individual lenses, thereby creating an electronic compound eye (cf. Franceschini et al., 1992). An alternative method is to use conventional CCD cameras in which the optics have been impaired by mounting sand-blasted glass screens in front of the lens so as to generate input images with focus-independent blur, prior to some coarse subsampling scheme.

In keeping with the incremental approach advocated previously, we have commenced our studies by adding just two photoreceptors to the set of sensors (bumpers

3 In conventional computer vision, image sizes of 512 × 512 (i.e., 262,144 pixels) are not considered particularly large.

4 The acceptance angle of a photoreceptor can be informally defined as twice its maximum incidence angle, where the maximum incidence angle is the largest angle, measured as eccentricity off the photoreceptor's visual axis ("direction of view"), at which an incoming ray of light can still have a significant effect.

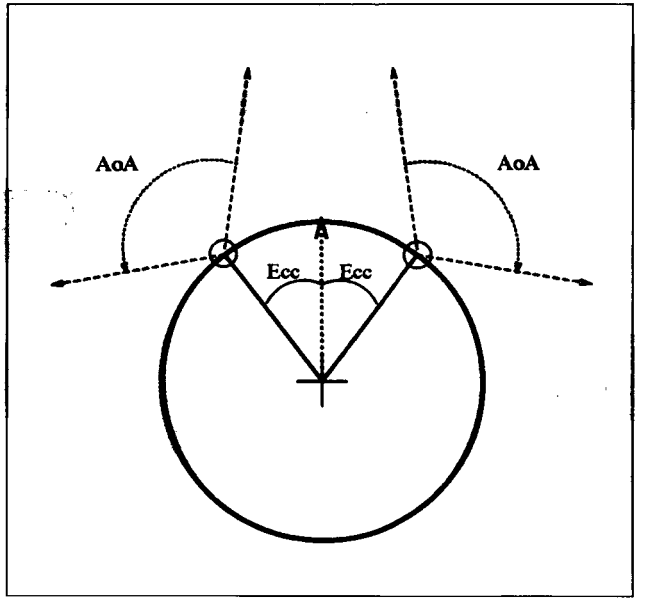


Figure 2
Angle of acceptance (*AoA*) and eccentricity (*Ecc*) for the two-photoreceptor robot. A top-down view of the robot and the relevant angles.

and whiskers) already described. Taking a cue from biological vision, the sensors are situated in positions that are bilaterally symmetrical about the robot's longitudinal midline.

Having only two receptors introduces manifest limitations on the classes of behaviors that can be expected to evolve in the robot. Assuming that the receptors sample largely distinct portions of the optic array, the only information the robot can access concerning its visual surroundings at any particular instant is likely to be limited to the raw data (the brightness levels recorded by the photoreceptors) and summary statistics such as the average brightness, or the difference between the two signals.

Nevertheless, the acceptance angles of the photoreceptors and their positions relative to the longitudinal axis can be varied under genetic control. The two receptors are constrained to have the same angle of acceptance, which is coded as a binary number represented as a bit-vector field in the robot's genome. A second bit-vector field in the genome governs the eccentricity of the photoreceptors, measured off the robot's longitudinal axis. Figures 2 and 3 illustrate these two angles.

Before we proceed to a detailed description of the experiments, we will elaborate on the particular type of neural networks and the genetic encoding used and will discuss the the simulation techniques employed to model the physics of the

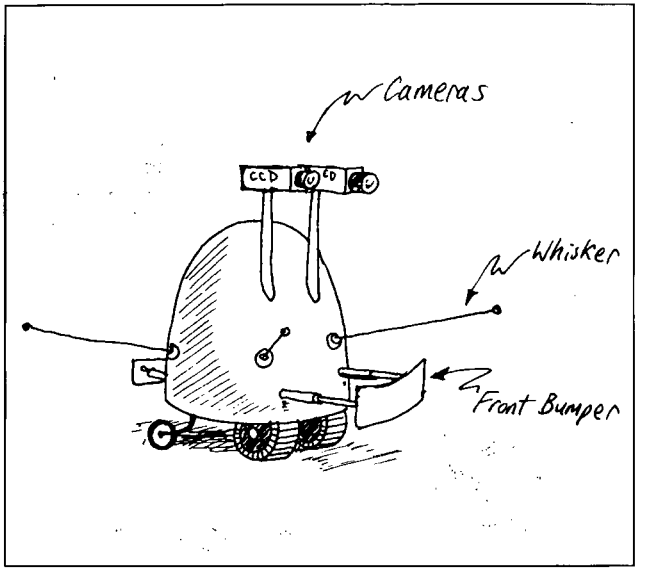


Figure 3
A cartoon of an appropriate robot: The angle of acceptance can be altered using zoom lenses. The eccentricity can be altered by rotating the cameras on their stalks.

robot's interaction with its environment and the computer graphics techniques used to simulate visual sensors.

3.3 The neural networks

As explained in section 2.5.2, we advocate the use of continuous real-valued networks with unrestricted connections and time delays between units. These are similar to analog circuits with real-valued signals continuously propagating through the connections. Our experience, as well as that of others (Beer & Gallagher, 1991), is that this sort of network can support a range of behaviors, depending on its exact couplings with the world, and can appear adaptive without using Hebbian-type weight changes or the like.

The particular networks used in the experiments have a fixed number of input units, one for each sensor, and a fixed number of output units, two for each motor. As all the units produce outputs in the range $[0.0, 1.0] \subset \mathbf{R}$, two units are used to give the motors a signal in the range $[-1.0, 1.0] \subset \mathbf{R}$. If the output signals from these four output units are labeled S_{o1}, to S_{o4} then, the left motor signal is given by $S_{o1}-S_{o2}$, and the right motor signal is given by $S_{o3}-S_{o4}$. As well as input and output units, each network will have some number of "hidden" units, which are not directly involved in either input or output. This number is not prespecified: The genotypes can vary in length, and increases or decreases in length correspond to the addition or deletion of units or links.

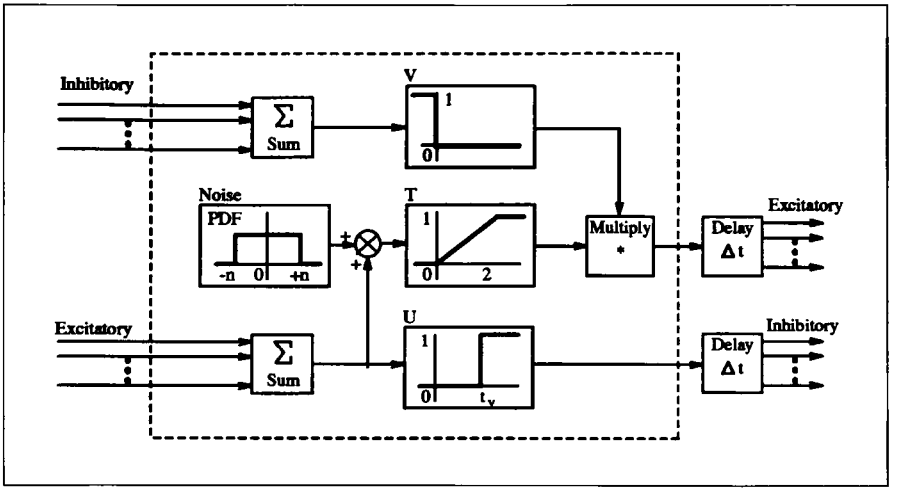


Figure 4
Schematic block diagram showing operations within a single model neuron. See text for further explanation.

Each unit is a noisy linear threshold device. Internal noise was added because we believed it would provide further useful and interesting dynamical properties. Any physical implementation of our networks would be likely to include naturally occurring noise anyway. A schematic diagram for the operations within each unit is shown in Figure 4.

There are separate excitation and inhibition channels within each unit. An *excitatory* connection is a weighted link joining the output of one unit to the input of another. A *veto* connection is a special infinitely inhibitory connection between two units. If there is a veto connection between units a and b, and a has any output on its inhibitory channel, then all excitatory outputs from b are turned off (though, in the current version, further veto outputs from b are not affected). The veto threshold is always much higher than the lower threshold for the excitatory ouptut. The veto mechanism is a crude but effective model of phenomena found in invertebrate nervous systems[5] and bears some similarities to the original proposals of McCulloch and Pitts (1943).

More formally, the excitation transfer function $\mathcal{T}$ takes the form: $\mathcal{T}(x) = 1$ if $x \geq t_u$; $\mathcal{T}(x) = 0$ if $x \leq t_l$; and $\mathcal{T}(x) = (x - t_l)/(t_u - t_l)$ otherwise, where t_l and t_u are lower and upper threshold levels. The veto output function $\mathcal{U}$ takes the form:

5 For example, feed-forward inhibition of the locust LGMD visual interneuron acts as a veto to prevent the LGMD from producing transient responses caused by delays in earlier processing. See, for example, Young (1989, pp. 77–78).

$\mathcal{U}(x) = 1$ if $x \geq t_v$, and 0 otherwise, where t_v denotes the veto threshold, and the veto input function $\mathcal{V}$ is: $\mathcal{V}(x) = 0$ if $x > 0$, and 1 otherwise.

Because there are separate excitation and inhibition channels, two connectivity matrices are required: a veto matrix V (where each element $v_{i,j}$ indicates whether unit i vetoes unit j) and an excitatory matrix W, with elements w_{ij}. Then, if $o_{ej}(t)$ and $o_{vj}(t)$ respectively denote the excitatory output and veto output from unit j at time t, and $N_j(t)$ denotes the internal noise injected to unit j at time t, the output channels from unit j can be expressed as:

$$o_{vj}(t) = \mathcal{U}\left(\sum_{\forall i} w_{ij} o_{ei}(t - \Delta t)\right)$$

$$o_{ej}(t) = \mathcal{V}\left(\sum_{\forall i} v_{ij} o_{vi}(t - \Delta t)\right) \cdot \mathcal{T}\left(N_j(t) + \sum_{\forall i} w_{ij} o_{ei}(t - \Delta t)\right)$$

Theoretical analysis reveals that small recurrent networks composed of such units are capable of exhibiting a range of useful properties, including acting as oscillators (Husbands, Harvey, & Cliff, 1993a, 1993b).

The genetic encoding specifies a controller network, including properties of individual units and the connections between them, as a linear string of characters drawn from a specified alphabet. Implicitly, our genetic encoding acts as a formal description language for our network architectures; it is now described in more detail.

3.4 The genetic encoding

To encode variably sized networks using a fixed number of sensors, we used two separate "chromosomes," one for the control network (variable length) and one for the factors governing the visual sampling (fixed length). When a new genotype is formed from two parents, crossover occurs on the vision and network chromosomes separately.

3.4.1 The network chromosome The genetic encoding used for the control network is illustrated in Figure 5. The network chromosome is interpreted sequentially. First the input units are encoded, each preceded by a marker. For each node, the first part of its gene can encode node properties such as threshold values; there then follows a variable number of character groups, each representing a connection from that node. Each group specifies whether it is an excitatory or veto connection, and then the target node is indicated by jump type and jump size. In a manner similar to that used in Harp et al. (1989), the jump type allows for both relative and absolute addressing. *Relative* addressing is provided by jumps forward or backward along the genotype order; *absolute* addressing is relative to the start or end of the genotype.

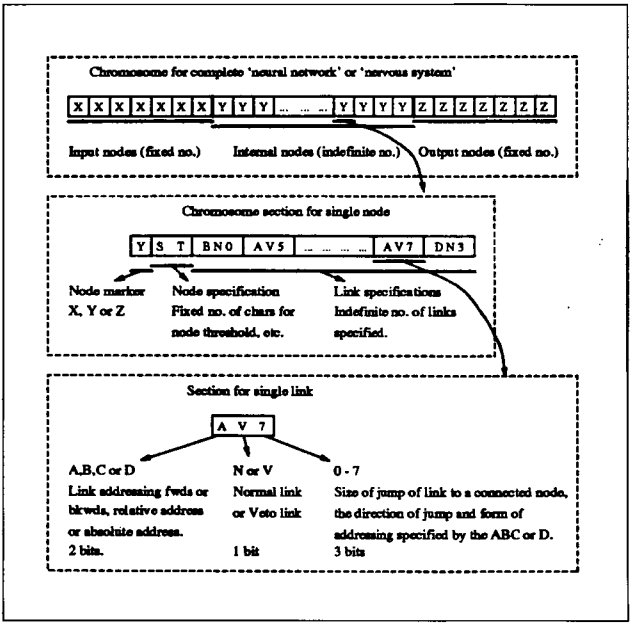


Figure 5
The genetic encoding scheme.

These modes of addressing mean that offspring produced by crossover will always be legal.

The internal nodes and output nodes are handled similarly, with their own identifying genetic markers. Clearly, this scheme allows for any number of internal nodes. The variable length of the resulting genotypes necessitates a careful crossover operator that exchanges homologous segments. In keeping with SAGA principles, when a crossover between two parents can result in an offspring of different length, such changes in length (although allowed) are restricted to a minimum.

3.4.2 The vision chromosome The details of the genetic coding of the acceptance angle α and the eccentricity β of the two photoreceptors is straightforward. In principle, the angles are constrained to the ranges $\alpha \in (0, \pi] \subset \mathbf{R}$ and $\beta \in [0, \pi/2] \subset \mathbf{R}$, but the use of a bit-vector genome forces a discretization of these ranges. Both angles are represented by four bits in the chromosome, giving a choice of $2^4 = 16$ discrete values for each angle, i.e., a total of $2^8 = 256$ configurations of α and β. If the integer values represented by the chromosome fields for α and β are i_α and i_β respectively (i_α, i_β both $\in \{0, 1, \ldots, 15\} \subset \mathbf{N}$), then $\alpha = (1 + i_\alpha)\pi/16$, and $\beta = i_\beta \pi/30$.

All experiments to date have involved evolving architectures that enable the robot to guide itself within a closed cylindrical room. The curved walls of the room are

black, whereas the floor and ceiling are white. Figure 6 illustrates output from the ray-tracing system in this environment. Figure 7 depicts the effects of varying α and β.

3.5 The physics: simulating movement

In our simulations, the continuous nature of the system was modeled by using fine time-slice techniques. At each time step, the sensor readings are fed into the neural network. The continuous nature of the networks is simulated by running them (synchronously updating all units' inputs and outputs for some number of iterations, with a stochastic variance in the number of iterations so as to counter distorting periodic effects) and then converting the outputs to motor signals. The new position of the robot is then calculated by using the appropriate kinematic equations. Using the wheel velocities, the motion is resolved into a rotation around one wheel plus a translation parallel to the velocity vector of the other. Standard Newtonian mechanics are used. However, the motion is not modeled as being wholly deterministic; noise is injected into the calculations. Collisions are handled as accurately as possible, using observations of the real system. The nature of the collision depends on speed and angle of incidence as well as the shape of the obstacle.

This type of simulation is not perfect. It can and will be made more accurate, but we believe it is realistic enough to allow our results to be taken seriously.

3.6 Simulating vision

The simulated robot was equipped with vision by embedding it within the SyCo vision simulator system described by Cliff (1991a). The SyCo simulator was developed for studying issues of visual processing for control of an airborne insect, but only minor alterations were required for our purposes: The agent's altitude was clamped at a constant value, because the robot is a wheeled vehicle traveling on a flat floor, and the visual sampling pattern, which is fixed in SyCo, had to be placed under genetic control.

The SyCo simulator synthesizes visual input by means of a computer graphics technique called *ray tracing* (see, for example, Glassner, 1989). This method involves integrating instantaneous point-sample estimates ("rays") of the relevant projection integrals, and so aliasing is a common problem. Put most simply, *aliasing* is a problem of insufficient samples being taken to give an accurate impression of the (visual) signal being sampled.

To limit the effects of aliasing, the SyCo code was configured to determine each photoreceptor's activity by averaging the readings from 16 rays per simulated receptor, distributed on a regular 4×4 regular grid across that receptor's visual field. This provides more accurate estimates of image brightness in the receptor's

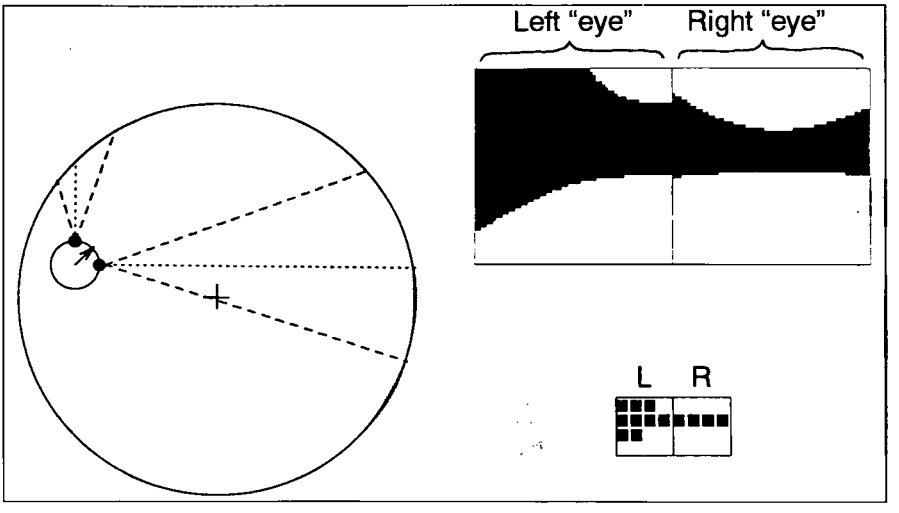


Figure 6
Illustration of the ray-tracing system. The left-hand figure shows the robot's position and orientation within the cylinder, which has black walls and white floor and ceiling. At upper right is a pair of relatively high-resolution images, traced from the robot's position inside the cylinder. The lower-right figure shows the two 4 × 4 images traced prior to averaging, with $\alpha = 1.571$ and $\beta = 0.628$. The final two photoreceptor brightness levels are derived by averaging the 4 × 4 images.

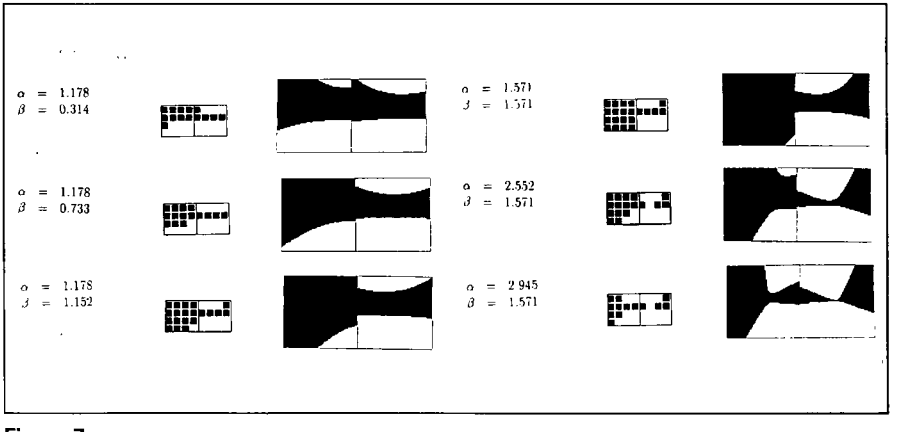


Figure 7
Varying α and β. For all the figures, the robot's position is the same as in Figure 6. For each combination of α and β, both the 4 × 4 and higher-resolution views are shown.

field of view. However, for two reasons—one pragmatic, the other theoretical—it is important to keep the number of rays per receptor relatively low. First, ray tracing is a computationally expensive process, so using fewer rays per receptor saves processing time. Second, real vision is not an arbitrary precision process. In vision, noise is inescapable, and noise effectively reduces a continuum of brightness levels to a small number of discrete ranges (e.g., Srinivasan, Laughlin, & Dubs, 1982). By limiting the number of rays per receptor, the precision of the brightness-value estimate is correspondingly reduced. The simulated robot must be able to cope with noisy limited-precision perception because that is all the real world has to offer.

4 Simulation Results

The primary goal in the experiments described here was to explore the possibility of evolving robots that could use their visual perception to perform tasks which would be difficult or impossible using only proximal sensing. As discussed already, the task chosen was to move to the center of a cylindrical arena, and stay there. We have described previously the evolution of controllers for tactile-only robots (Harvey et al., 1993).

The evolutionary process starts with a population of genotypes; we have used populations of size 60. Initially, all the genotypes in the population are random character strings. On every generation, each genotype is evaluated and assigned a fitness score. The genotypes are then interbred, with mutation and crossover according to SAGA principles, thereby creating a new population. This process continues for a prespecified number of generations. Although our genetic encoding allows for the weights, delays, and thresholds for each unit to be determined separately, the results reported here are from experiments in which all these values were fixed at the same constant for all units throughout the course of evolution. All delays and actual (i.e., nonzero) weights were set to unity; the threshold values were: $t_l = 0.0$; $t_u = 2.0$; $t_v = 0.75$; and the noise parameter was $n = 0.1$. Thus the only aspects of the network under genetic control were the number of hidden units and the specific connections between units.

The evaluation of each genotype involves decoding the chromosomes to create a simulated robot and then testing the robot a number of times (we use eight tests per genotype). On each test, the robot is positioned at a random orientation and position in the arena (with a bias toward positions near the wall, so that it starts at least two-thirds of the radius of the arena away from the center), and the orientation was also randomly chosen. The robot is then allowed a fixed amount of simulated time during which its behavior is rated according to an evaluation function $\mathcal{E}$. $\mathcal{E}$ varies according to the behavior we want the population of robots to exhibit: A discussion

of some of the earlier $\mathcal{E}$ functions we used is provided in an earlier publication (Cliff, Husbands, & Harvey, 1993). At the end of the eight tests, the *lowest* value of $\mathcal{E}$ scored on the tests is used as the robot's fitness value in the reproductive phase: this ensures robust solutions. (If the best or average $\mathcal{E}$ value is used, it can be deceptively high.)

Figure 8 shows the full network for the most fit individual genotype evolved by maximization of the evaluation function:

$$\mathcal{E} = \sum_{\forall t} \exp(-s|\mathbf{r}(t)|^2)$$

where $\mathbf{r}(t)$ is the two-dimensional vector from the robot's position to the center point on the floor of the circular arena at time t, and $\forall t$ denotes the duration of the evaluation test (the sum is essentially a discrete approximation to a temporal integral). Put most simply, the more time the robots spend at or near the center of the arena, the higher they are rated. The value s is a scale factor that ensures the robot collects no score if it is near the wall of the arena. Note that in Figure 8, the network has three hidden units, one more than the maximum given to the initial random genotypes. The vision chromosome for this individual specified an acceptance angle of 45 degrees and an eccentricity of 6 degrees.

Under this evaluation function, the optimal behavior is, from a random initial starting position, to move toward the center of the arena as quickly as possible and then, when at the center, to stay there. As can be seen from Figure 9, this is precisely what the robot does when controlled by the network of Figure 8.

The robot was evaluated using $\mathcal{E}$ over a duration of 100 time steps. Hence, the maximum score possible would be 100 if the robot spent all its time at the exact center of the arena. However, because the robots always started some distance from the center, the maximum score possible, in practice, is somewhere between 75 and 85, depending on the initial orientation and location and on the noisy interactions between the robot and its world.

Figure 10 shows the best, average, and worst of the eight $\mathcal{E}$ scores for the most fit individual in the population at each generation. Because SAGA maintains a high degree of convergence in the population, this can be treated informally as the evolutionary history of the particular network shown in Figure 8.

Selection was rank-based, with a quadratic used to convert ranking into expected contribution to the next generation. The first i members of a population size p have between them a quota proportional to $1/\sqrt{i}$. When $p = 60$, this means that the first 16 all have above-average quotas, and the very first contributes an expected $\sqrt{60} \simeq 7.75$ to the gene pool for the next generation. This is, compared to standard genetic algorithms, abnormally high selection, which the rank-based method maintains indefinitely. The mutation rate was set at an expected 0.9 bits flipped per

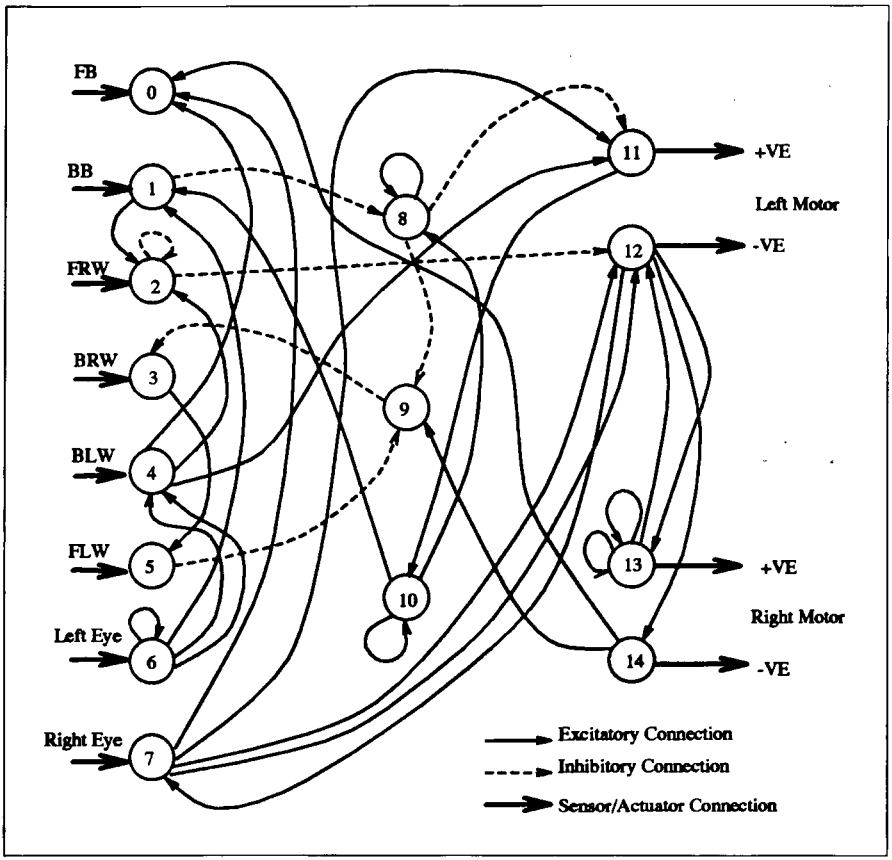


Figure 8
An evolved control network. In the left-hand column are units initially designated as input units: *FB* = front bumper; *BB* = back bumper; *FRW* = front right whisker; *BRW* = back right whisker; *BLW* = back left whisker; *FLW* = front left whisker. Right-hand column shows output units, which are paired and differenced to give two motor signals in the range $[-1, 1]$ from four neuron outputs in the range $[0, 1]$. Center column shows hidden units.

genotype. We also used elitism, whereby the genotype for the most fit individual in each generation is copied over to the next generation without change.

It was our intention to impose as little structure as possible on the control networks, but it is necessary to designate some units as *input* units (receiving activity from the robot's tactile or visual sensors) and some as *output* units (the activity level of which determines the output of the two drive motors). Units that are neither input nor output are referred to as *hidden*. As will be seen later, the evolutionary process can blur these distinctions.

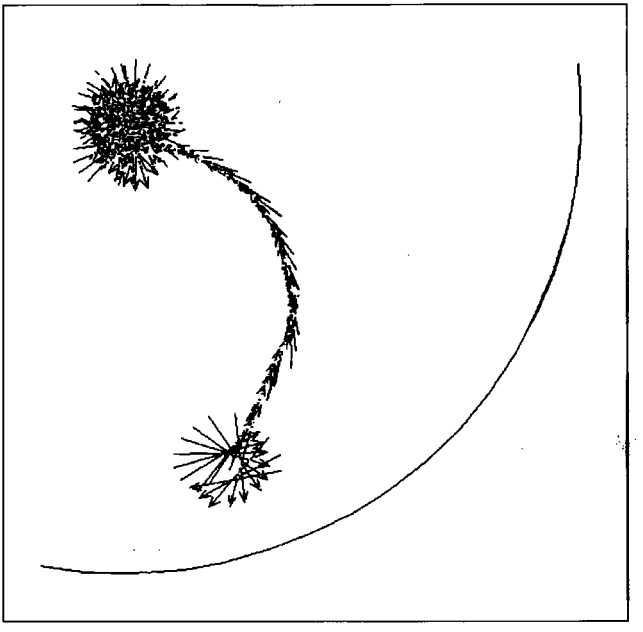


Figure 9
Typical behavior of an evolved controller, illustrated in a plan view of the robot's position within the circular arena at each step in its 100-step lifetime. The robot's position at each time step is shown by an arrow; the midpoint of the arrow shaft is the center of the robot, and the length of the shaft is the same as the robot's diameter. The robot starts near the edge of the arena, moves to the center, and then spins on the spot. The tip of the arrow shows the front of the robot, which is not necessarily the direction of travel: Although in this case the robot is moving forward, it can travel in reverse.

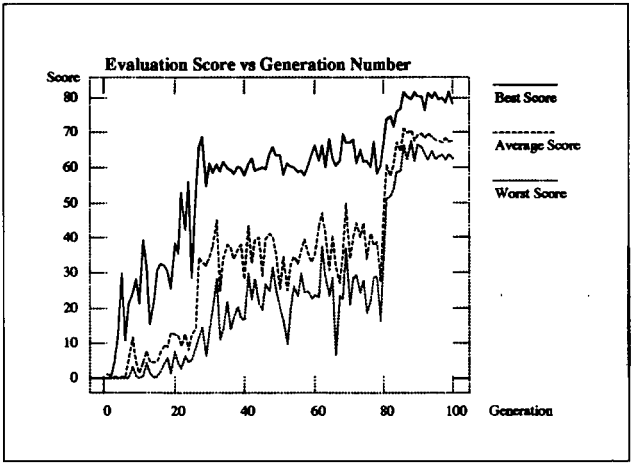


Figure 10
Evolutionary history of the network shown in Figure 8. See text for further details.

The initial random genotypes are created to encode for networks with all the necessary input and output units and either one or two hidden units. Because we use SAGA, the genotypes can vary in length: Longer genotypes can arise, the increase in length corresponding to more connections or extra hidden units; but such increases in the size of the network will, in general, be carried forward to subsequent gener-

ations if they achieve higher fitness ratings in the evaluation process. In this sense, more complex networks will develop in an *incremental* fashion. When using these techniques, it may eventually be necessary to introduce into the evaluation function costs that penalize redundant nodes or connections, but we have not done so in the work reported here.

For each $\mathcal{E}$ we have studied, we set up eight separate random populations and allowed each to evolve for 100 generations.[6] When this was complete, we took the genotype with the highest fitness from each population and analyzed its performance. Typically, in each batch of eight populations, three to five of them had improved only moderately on the performance of the initial random genotypes, whereas the remainder were scoring close to maximum fitness.

We have described the analysis process in greater depth elsewhere (Cliff, Husbands, & Harvey, 1992; Husbands et al., 1993b). For the current discussion, it is sufficient to note that the analysis process involved decoding the genotype to produce a circuit diagram for the network. Qualitative techniques, inspired by standard practices in neuroethology, were applied to identify and eliminate redundant aspects of the network. Redundancy is typically introduced where structures that were beneficial, or at least not actually harmful, in earlier generations are retained while their function is subsumed or superceded by structures that evolve later—that is, we eliminated vestigial structures from the networks. Then, by monitoring the values of appropriate variables (e.g., activation values) while the robot interacts with its environment, we were able to establish causal explanations for the interactions between units in the network. Furthermore, we applied quantitative analytical techniques using a mix of theoretical and empirical methods. The theoretical work involved identifying significant feedback loops in the network and analyzing them under steady-state conditions. Empirical observations were then made, to validate the predictions from theoretical analysis, using observations that employ analytical visualization techniques which treat the robot and environment as coupled dynamical systems.

Analyzing the network in Figure 8 revealed that this controller was performing guidance using *only* its visual sensors: The controller developed the ability to monitor visual input to detect and avoid imminent collisions with the walls. Once such a behavioral competence has evolved, the tactile sensors and their associated input units are rendered redundant. Moreover, the opportunistic nature of evolution is such that it is not uncommon for the redundant input units to be "re-allocated" by chance mutations, so they then serve as visual processing units. Though it is not immediately obvious, this has occurred in the network of Figure 8: The final reduced network, with the redundant channels eliminated, is shown in Figure 11.

6 Typically, it takes approximately 24 hours on a Sun SPARC–2 workstation to evolve one population; we evolved the eight populations in parallel, on eight separate workstations.

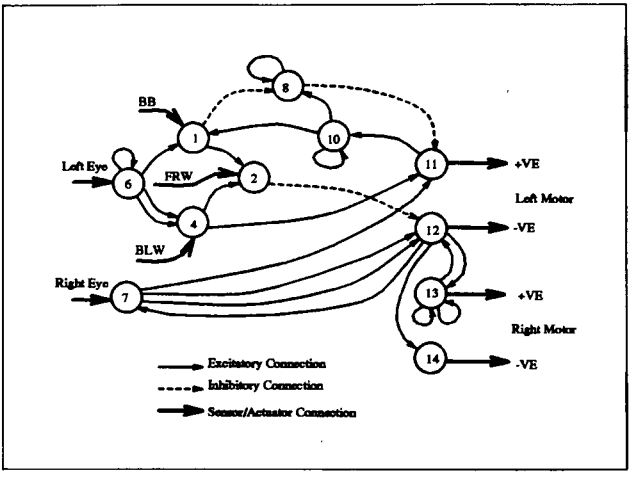


Figure 11
Final reduced network. (*BB* = back bumper; *FRW* = front right whisker; *BLW* = back left whisker.

As can be seen, one unit that was involved originally in tactile input is now serving as a second-order visual processing unit. (For further details of the analysis of this network, see Cliff et al., 1992.)

Other controllers, evolved under the same conditions as this one, have been analyzed in more depth. In the work reported by Husbands et al. (1993b), we demonstrate that one particular controller has evolved that will operate successfully in a wide range of circular arenas, despite having been evolved in a single environment that was fixed throughout evolution.

Further Work

The encouraging results from our early experiments have revealed a strong potential for further work. There are a number of fairly obvious directions in which the work can be taken, such as applying these techniques to other control tasks, other model neurons, more advanced types and combinations of sensors and effectors, different selection pressures, lifetime learning mechanisms, and so on. None of these would necessarily require significant revisions to the experimental methods described herein.

However, on the basis of our work thus far, we believe there are two topics that require attention before significant progress can be made: The first involves methods for genetic encoding, and the second involves ending the reliance on simulation. Both are discussed in more detail next.

5.1 New genetic encodings

The aspect of our work that we believe is most in need of revision is the genetic encoding of the control networks used. One basic lack in our current encoding scheme is the ability to encode repeated structures in a succinct way. A mechanism analogous to subroutining in programming languages is required to give concise genetic encodings of controllers that have a high degree of regularity in their structure.

Furthermore, in the current scheme, the network chromosome is essentially a specification of the wiring diagram for the network. This instructionist view of genes as programs (cf. Miller et al., 1989) is not one with which we agree. Rather, we take the view that the genotype for an individual acts as a set of constraints on the possible developmental path of that individual and, for this reason, we are currently considering the introduction of artificial developmental processes.

Finally, we would like to escape from the (arbitrary) division of having separate chromosomes, one for sensor morphology and one for the control network. It may well be that the concurrent evolution of sensorimotor morphology and controller is actually hampered by this division: Crossover might have a less disruptive effect if genes encoding for morphological features are positioned close on the genome to genes encoding for the control structures that take advantage of those morphological features. This might allow for a closer integration of morphology and associated control than is possible in our current scheme.

5.2 Avoiding simulation

To our knowledge, the only successful application of (straightforward) genetic algorithms to the evolution of neural network controllers for a real robot is described in the report by Lewis, Fagg, and Solidum (1992). Lewis and co-workers used a six-legged walking robot called *Rodney*, evolving circuitry for connecting pregiven oscillators while selecting for stable gait patterns. The evaluations involved human observers monitoring and evaluating the performance of the *actual* hardware, the robot was not simulated. However, while the use of real hardware for evaluation is admirable, the reliance on manual evaluation is highly labor-intensive. In more complex or temporally extended tasks, automatic evaluation is a necessity but, as far as we are aware, all previous automatic evaluation schemes have relied on simulations that have sacrificed accuracy for speed and hence have produced results of questionable utility. We have developed a hardware approach that is intended to combine the positive aspects of these two approaches—that is, to allow for extended automatic monitoring of real hardware. Because careful simulations are time-consuming to write, validate, and run, it would be attractive to minimize any reliance on simulation. That is, we need to find hardware solutions to the problems of having to evaluate repeatedly a large number of individuals over tens or hundreds of generations.

Ideally, we would like to have a population of robots, each with fast parallel computation on-board and dynamically variable sensorimotor morphologies, in order that they could have a genotype downloaded automatically, either via a wireless communication link or by temporarily (and autonomously) docking with a downloader unit. Although such robots may be feasible in principle, in practice limited resources require that we find a less costly solution.

A significant issue is whether all the computational processing should be done on-board the robot, or off-board via some link to more powerful stationary computers. Both these choices have associated negative factors—namely, the size and weight of on-board computation or the problems of radio links and tangled umbilical cables.

Thus, for experimental purposes we have devised a third method that provides a miniature robot with visual processing that can roam freely through an environment set up by the experimenters. The environment could be a "toytown," although the word *toy* here refers only to size. The robot is in a real world, with real-world vision problems.

A sketch of the the toytown system is shown in Figure 12. A gantry is set up above a flat surface, with a horizontal girder able to move west and east by means of a stepper motor, providing the x coordinates of the robot. Along the girder, another stepper motor allows movement of a platform north and south (the y coordinates). From the platform, a CCD video camera is fixed pointing vertically downward. A custom-built 50-Hz frame grabber digitizes low-resolution images from the camera. A mirror is mounted below the camera and at roughly 45 degrees to the camera's optical axis: The mirror can be rotated about the optical axis of the camera while keeping the angle constant, rather like an inverted periscope. The camera does not rotate, and so its cables do not become twisted. The camera and mirror can be moved together via its supporting platform in the x and y dimensions. Vertical movement relative to the platform can also be provided to give the z dimension. The mirror itself can be considered the body of the sighted artificial creature, which can move through the environment provided for experimental purposes, and tactile sensors can be mounted on the mirror or camera housing to allow for collision detection and the like.

The maximum available field of view is determined by the optics of the camera and the geometry of the mirror (in the limit, a conical mirror gives a 360-degree horizontal field of view), but software sampling (under "genetic" control) can provide any number of "virtual" pixels or photoreceptors facing any specified direction within the limits of the hardware. Rotation of this visual field about the vertical axis can be effected in software, as can any number of strategies for sampling the visual field (cf. Cliff & Bullock, 1992). A system of stepper motors, sprockets, and chains provide an accuracy of movement of plus or minus 1 mm. Touch sensors around the mirror housing complete the so-called body of the robot. The robot's control

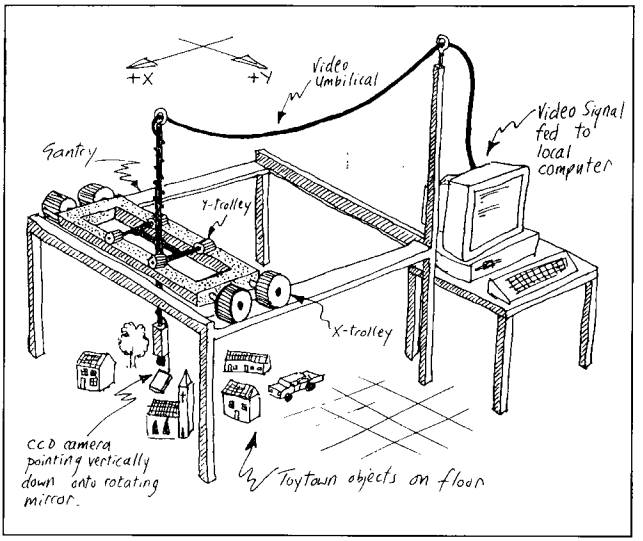


Figure 12
The toytown gantry system. See text for further details.

network is simulated off-board on a computer. The sensory inputs are fed into the controller via an umbilical cable and interfacing cards. In a similar way, the controller sends motor commands to the various actuators.

The body of the robot is only the size of the mirror plus touch sensors and, subject only to its attachment to the camera above, and hence to the gantry, it can be moved anywhere within the volume bounded by the gantry. The environments constructed in this space can be suitably small and easily altered. In this way, all the real-world characteristics of moving around in a noisy visual environment are retained, with a number of advantages for experimental purposes over a wheeled ground-based autonomous robot:

- There are no problems with tangled umbilicals, and on-board power supplies and computers are not an issue.
- The environment is easily changed.
- "Robot time" can be slowed down to a rate appropriate to computational resources. As cognitive processing becomes more computationally demanding, the speed of movement of the creature and other dynamical objects in the environment can be made as slow as is desired.
- The highly controllable nature of the apparatus means that experiments are repeatable, and very long runs can be achieved without any human intervention. Hence, for each generation, each member of the population can be evaluated without recourse to simulation.

A succession of tasks of increasing complexity can be set for such a robot. Automatic evaluations for each task allow for a succession of tests, and for the evolutionary process as a whole, to continue without immediate human intervention. A possible succession of tasks would be:

- Movement toward an object (a prominent dark mark, perhaps).
- Rotation to face a moving object.
- Avoidance of obstacles.
- Movement between two objects.
- Movement centrally along a striped corridor.
- Identification of, and movement through, doorways.
- Exploratory movement within a simple maze.
- Identification of particular locations within such a maze or environment, and return to them after exploration.
- Development of place recognition by navigation through the environment between specified points via self-selected intermediary places.
- Navigation and interaction with a dynamical world.
- Performance of the preceding tasks but subject to arbitrary polarity reversal of the motor outputs.

The outputs from the controller provide signals to the motor drives (with the deliberate addition of noise, if desired) that effectively allow the robot to move continuously and freely in this world. The robot is, for all intents and purposes, autonomous. However, although it does not know its absolute position and orientation, this information is always available to the experimenters and is extremely useful for automatic fitness evaluation, repeatability, repositioning, and so on. Our toytown robot has been built and is now in use (Figs. 13 and 14).

The toytown environment bears some similarities to the tinytown environment at Rochester (Nelson, 1991). However, the latter has a camera pointing down that can move only in two dimensions, giving the equivalent of low-flying aerial photographs. In contrast, the toytown robot has a rotational degree of freedom and can travel in and among the objects of a three-dimensional world, with a horizontal field of view manipulable between 0 and 360 degrees, depending on the mirror used.

5.3 Biological modeling

One application of our methods, which may require working wholly in simulation, is the study of particular biological systems. Although research in artificial autonomous agents often shows signs of heavy biological inspiration, we are keen to see whether we can use our methods to produce models of direct use to biologists. As was

Figure 13
The toytown robot is suspended on a vertical pole beneath the moving gantry. One stepper motor on the platform above the robot can raise and lower it, in the z direction. A second stepper motor mounted at one end of the cross-gantry controls movement in the y direction. A third stepper motor on the end of the frame moves the whole gantry in the x direction via chains. An umbilical cord carries motor signals to the motors and sensor and camera signals to the computers controlling the system.

mentioned at the start of this article, a biologist proposing a model sufficient for some aspect of sensorimotor coordination in a particular species is effectively engaging in an act of creative design, subject to clues and constraints arising from experimental data. Just as our methods help (partially) automate the design process in constructing autonomous mobile robots, so they may help partially automate the design process in proposing biological models.

Such an approach is not without precedent: In addition to the work of Beer and Gallagher (1991) discussed earlier, which may further the models proposed in Beer's earlier publication (1990), Stork, Jackson, & Walker (1992) discuss using a genetic algorithm to simulate the evolution of neural circuitry underlying an escape behavior in crayfish. They demonstrate that a particular connection in the biological network

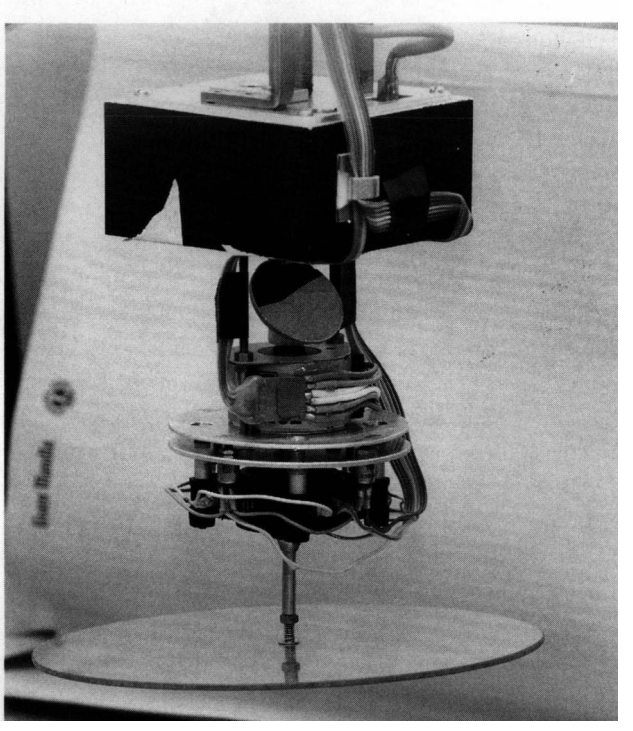


Figure 14
A close-up of the toytown robot. The black box contains a CCD camera pointing down at the mirror, which is at a 45-degree angle to vertical. The mirror can be rotated about a vertical axis, by the stepper motor beneath, through 360 degrees. The large plastic disc at the base is attached to a joystick system and acts as a touch sensor that discriminates eight possible directions of contact with an obstacle.

can be explained as a result of preadaptation, in that the circuit was used for swimming behaviors before it came to serve the escape function. While these investigators consider only the evolution of controller circuitry (i.e., they do not evolve the motor morphology), their experimental methods include a genetic encoding more sophisticated and biologically inspired than ours and a continuous dynamical neuron model based on the Hodgkin-Huxley equations.

To allow morphology to be evolved concurrently with controller circuitry, it would be necessary to set up a simulation that is sufficiently close to the real animal's *merkwelt* in the context of the particular behavior under study such that application of a selective pressure may lead to selection of individuals which exhibit the behavior required. An incremental evolution scheme would almost definitely be required. We currently are working on employing the toytown robot for the study of characteristic orientation flights performed by bees and wasps.

6 Conclusions

There is no evidence to suggest that humans are good at designing systems which involve many emergent interactions between many constituent parts, but robust control systems for robots may well fall under this classification. Artificial evolution seems a good way to move forward, and it is advocated in this article.

To support our claims, we have presented simulation results from experiments in evolving sensorimotor control architectures for mobile robots. Using parallel networks of relatively constrained processing units and simple evaluation functions, we have been able to evolve visual control architectures. We consider it significant that we were able to evolve visually guided controllers, even when the evaluation function $\mathcal{E}$ does not explicitly require monitoring visual inputs: The evolutionary processes capitalized on the availability of distal sensory information, without explicit coercion. We see no reason why similar results should not be possible when other combinations of sensory modalities are explored.

The results have demonstrated the feasibility of the approach, but the computational costs of simulating vision have led us to develop a method that allows for a mix of real vision and evolutionary methods, using cheap readily available hardware. The toytown project is at an early stage, but our current results are sufficiently promising that we do not foresee significant problems in transferring from simulations to real hardware.

Acknowledgments

The authors thank Tony Simpson, Martin Nock, Jerry Mitchell, and Harry Butterworth for the engineering design and construction work on the robot. Many thanks also to Geoffrey Miller for his valuable comments on an earlier version of this article.

Inman Harvey has been supported by the U.K. Science and Engineering Research Council. The toytown robot was funded by a University of Sussex research and development grant.

References

Ackley, D. H., & Littman, M. L. (1992). Interactions between learning and evolution. In C. G. Langton, J. D. Farmer, S. Rasmussen, & C. Taylor (Eds.), *Artificial life II: Proceedings of the Santa Fe Conference, February 1990, XI.* Reading, MA: Addison-Wesley.

Altman, J. S., & Kien, J. (1989). New models for motor control. *Neural Computation, 1*, 173–183.

Arbib, M. A. (1989). *The metaphorical brain 2.* Wiley.

Barhen, J., Dress, W., & Jorgensen, C. (1987). Applications of concurrent neuromorphic algorithms for autonomous robots. In R. Eckmiller & C. Malsburg (Eds.), *Neural computers*. New York: Springer-Verlag.

Beer, R. D. (1990). *Intelligence as adaptive behavior: An experiment in comptational neuroethology*. New York: Academic Press.

Beer, R. D. (1992). *A dynamical systems perspective on autonomous agents* (Technical Report CES-92-11). Cleveland, OH: Case Western Reserve University.

Beer, R. D., & Gallagher, J. C. (1991). Evolving dynamical neural networks for adaptive behaviour. *Adaptive Behavior, 1*(1), 91–122.

Braitenberg, V. (1984). *Vehicles: Experiments in synthetic psychology*. Cambridge, MA: MIT Press/Bradford Books.

Brooks, R. A. (1985). *A robust layered control system for a mobile robot* (A.I. Memo 864). Cambridge, MA: MIT/Artificial Intelligence Laboratory.

Brooks, R. A. (1986). *Achieving artificial intelligence through building robots* (A.I. Memo 899). Cambridge, MA: MIT/Artificial Intelligence Laboratory.

Brooks, R. A. (1990a). *The behavior language* (A.I. Memo 1227). Cambridge, MA: MIT/Artificial Intelligence Laboratory.

Brooks, R. A. (1990b). Challenges for complete creature architectures. In J.-A. Meyer & S. W. Wilson (Eds.), *From animals to animats: Proceedings of the First International Conference on Simulation of Adaptive Behavior (SAB90)*. Cambridge, MA: MIT Press/Bradford Books.

Brooks, R. A. (1991). Intelligence without representation. *Artificial Intelligence, 47*, 139–159.

Brooks, R. A. (1992). Artificial life and real robots. In F. J. Varela & P. Bourgine (Eds.), *Toward a practice of autonomous systems: Proceedings of the First European Confernce on Artificial Life (ECAL91)*. Cambridge, MA: MIT Press/Bradford Books.

Chapman, D. (1987). Planning for conjunctive goals. *Artificial Intelligence, 32*(3), 333–377.

Cliff, D. T. (1991a). The computational hoverfly; a study in computational neuroethology. In J.-A. Meyer & S. W. Wilson (Eds.), *From animals to animats: Proceedings of the First International Conference on Simulation of Adaptive Behavior (SAB90)*. Cambridge, MA: MIT Press/Bradford Books.

Cliff, D. T. (1991b). Computational neuroethology: A provisional manifesto. In J.-A. Meyer & S. W. Wilson (Eds.), *From animals to animats: Proceedings of the First International Conference on Simulation of Adaptive Behavior (SAB90)*. Cambridge, MA: MIT Press/Bradford Books.

Cliff, D. T., & Bullock, S. G. (1992). Adding "foveal vision" to Wilson's animat. *Adaptive Behavior, 2*(1), 49–72.

Cliff, D. T., Husbands, P., & Harvey, I. (1992). *Analysis of evolved sensory motor controllers* (Technical Report CSRP 264). Brighton, England: University of Sussex School of Cognitive and Computing Sciences.

Cliff, D. T., Husbands, P., & Harvey, I. (1993). Evolving visually guided robots. In J.-A. Meyer, H. Roitblat, & S. Wilson (Eds.), *Proceedings of the Second International Conference on Simulation of Adaptive Behavior (SAB92)*. Cambridge, MA: MIT Press/Bradford Books.

Connell, J. H. (1990). *Minimalist mobile robotics: A colony-style architecture for an artificial creature.* New York: Academic.

de Garis, H. (1990). Genetic programming: Building artificial nervous systems using genetically programmed neural network modules. In B. W. Porter & R. J. Mooney (Eds.), *Proceedings of the Seventh International Conference on Machine Learning.* San Mateo, CA: Morgan Kaufmann.

Franceschini, N., Pinchon, J.-M., & Blanes, C. (1992). From insect vision to robot vision. *Philosophical Transactions of the Royal Society of London [B], 337,* 283–294.

Glassner, A. S. (Ed.). (1989). *An introduction to ray tracing.* London: Academic.

Goldberg, D. E. (1989). *Genetic algorithms.* Reading, MA: Addison-Wesley.

Harp, S. A., Samad, T., & Guha, A. (1989). Towards the genetic synthesis of neural networks. In J. D. Schaffer (Ed.), *Proceedings of the Third International Conference on Genetic Algorithms.* San Mateo, CA: Morgan Kaufmann.

Harvey, I. (1992a). The SAGA cross: The mechanics of crossover for variable-length genetic algorithms. In R. Männer & B. Manderick (Eds.), *Parallel problem solving from nature, 2.* New York: Elsevier/North-Holland.

Harvey, I. (1992b). Species adaptation genetic algorithms: A basis for a continuing SAGA. In F. Varela & P. Bourgine (Eds.), *Toward a practice of autonomous systems: Proceedings of the First European Conference on Artificial Life (ECAL91).* Cambridge, MA: MIT Press/Bradford Books.

Harvey, I. (1993). Evolutionary robotics and SAGA: The case for hill crawling and tournament selection. In C. Langton (Ed.), *Artificial Life 3: Proceedings of the Santa Fe Conference, XVI.* Reading, MA: Addison-Wesley. In press.

Harvey, I., Husbands, P., & Cliff, D. (1993). Issues in evolutionary robotics. In J.-A. Meyer, H. Roitblat, & S. Wilson (Eds.), *Proceedings of the Second International Conference on Simulation of Adaptive Behavior (SAB92).* Cambridge, MA: MIT Press/Bradford Books.

Holland, J. H. (1975). *Adaption in natural and artificial systems.* Ann Arbor: University of Michigan Press.

Horswill, I. D. (1993). A simple, cheap, and robust visual navigation system. In J.-A. Meyer, H. Roitblat, & S. Wilson (Eds.), *Proceedings of the Second International Conference on Simulation of Adaptive Behavior (SAB92).* Cambridge, MA: MIT Press/Bradford Books.

Husbands, P., & Harvey, I. (1992). Evolution versus design: Controlling autonomous robots. In *Integrating perception, planning, and action: Proceedings of the Third Annual Conference on Artificial Intelligence, Simulation and Planning.* New York: IEEE Press.

Husbands, P., Harvey, I., & Cliff, D. T. (1993a). Analysing recurrent dynamical networks evolved for robot control. In *Proceedings of the Third IEE International Conference on Artificial Neural Networks (IEE-ANN93).* London: IEE Press.

Husbands, P., Harvey, I., & Cliff, D. T. (1993b). *Circle in the round: State space attractors for sighted robots.* Manuscript submitted for publication.

Kauffman, S. (1989). Adaptation on rugged fitness landscapes. In D. L. Stein (Ed.), *Lectures in the sciences of complexity.* Reading, MA: Addison-Wesley.

Kerszberg, M., & Bergman, A. (1988). The evolution of data processing abilities in competing automata. In R. M. J. Cotterill (Ed.), *Computer simulation in brain science.* Cambridge, England: Cambridge University Press.

Koza, J. (1992). *Genetic programing: On the programming of computers by means of natural selection.* Cambridge, MA: MIT Press.

Koza, J. R. (1990). *Genetic programming: A paradigm for genetically breeding populations of computer programs to solve problems* (Technical Report STAN-CS-90-1314). Stanford, CA: Department of Computer Science, Stanford University.

Lewis, M. A., Fagg, A. H., & Solidum, A. (1992). Genetic programming approach to the construction of a neural network for control of a walking robot. In *Proceedings of the IEEE International Conference on Robotics and Automation*, Nice, France.

Maturana, H. R., & Varela, F. J. (1987). *The tree of knowledge: The biological roots of human understanding.* Boston, MA: Shambhala Press.

McCulloch, W. S., & Pitts, W. (1943). A logical calculus of the ideas immanent in nervous activity. *Bulletin of Mathematical Biophysics, 5*, 115–133.

McFarland, D. (1990). What it means for robot behaviours to be adaptive. In J.-A. Meyer & S. W. Wilson (Eds.), *From animals to animats: Proceedings of the First International Conference on Simulation of Adaptive Behavior (SAB90).* Cambridge, MA: MIT Press/Bradford Books.

Miller, G. F., Todd, P. M., & Hegde, S. U. (1989). Designing neural networks using genetic algorithms. In J. D. Schaeffer (Ed.), *Proceedings of the Third International Conference on Genetic Algorithms.* San Mateo, CA: Morgan Kaufmann.

Moravec, H. (1983). The Stanford Cart and the CMU Rover. In *Proceedings of the IEEE, 71*, 872–884.

Muhlenbein, H., & Kindermann, J. (1989). The dynamics of evolution and learning —towards genetic neural networks. In R. Pfeifer, Z. Schreter, F. Fogelman-Soulie, & L. Steels (Eds.), *Connectionism in perspective.* New York: Elsevier/North-Holland.

Nelson, R. C. (1991). Visual homing using an associative memory. *Biological Cybernetics, 65*, 281–291.

PRANCE (1991). *Perceptive robots: Autonomous navigation and cooperation through evolution.* Unpublished research proposal, PRANCE consortium: University of Sussex, Cap Gemini Innovation, École Normale Supérieure (Paris) and Université Libre de Bruxelles.

Sejnowski, T. J., Koch, C., & Churchland, P. S. (1988). Computational neuroscience. *Science, 241*, 1299–1306.

Srinivasan, M. V., Laughlin, S. B., & Dubs, A. (1982). Predictive coding: A fresh view of inhibition in the retina. *Proceedings of the Royal Society of London: Biological Sciences, 216*, 427–459.

Stork, D., Jackson, B., & Walker, S. (1992). 'Non-optimality' via pre-adaptation in simple neural systems. In C. Langton, C. Taylor, J. Farmer, & S. Rasmussen (Eds.), *Artificial life II: Proceedings of the Santa Fe Conference, February 1990, XI.* Reading, MA: Addison-Wesley.

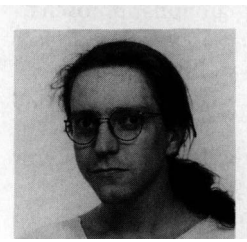
van Gelder, T. (1992). *What might cognition be if not computation?* (Technical Report 75). Bloomington, IN: Indiana University, Department of Cognitive Science.

Viola, P. (1988). *Mobile robot evolution.* Unpublished bachelor's thesis, MIT, Cambridge, MA.

Wilson, S. W. (1985). Knowledge growth in an artificial animal. In J. J. Grefenstette (Ed.), *Proceedings of an International Conference on Genetic Algorithms and Their Ap-*

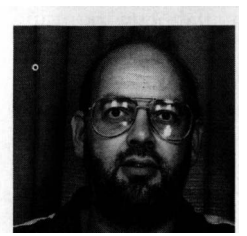
plications (ICGA85). Pittsburgh, PA: Hillsdale, NJ: Lawrence Erlbaum Associates.
Young, D. (1989). *Nerve cells and animal behaviour.* Cambridge, England: Cambridge University Press.

About the Authors



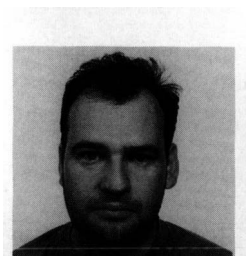
Dave Cliff

Dave Cliff has received his bachelor of sciences degree in computer science from the University of Leeds and master of arts and doctor of philosophy degrees in cognitive science from the University of Sussex. His research interests include mechanisms underlying the generation of adaptive behaviors, particularly visual sensorimotor coordination, in both animals and artificial autonomous agents.



Inman Harvey

Inman Harvey was awarded a master of arts degree in mathematics and philosophy as well as a postgraduate certificate in social anthropology from the University of Cambridge and a master of sciences degree in knowledge-based systems from the University of Sussex. His research toward obtaining his doctoral degree in cognitive science at the University of Sussex has centered on the theoretical foundations of the artificial evolution of behavior.



Phil Husbands

Phil Husbands received a bachelor of sciences degree in theoretical physics from the University of Manchester, a master of sciences degree in computer science from South Bank Polytechnic, and a doctorate in applied artificial intelligence from the University of Edinburgh. His research interests concern evolutionary algorithms, robotics, neural networks, and mechanisms for generating low-level adaptive behaviors. Dr. Husbands is a lecturer in artificial intelligence.