Evolving a Nervous System of Spiking Neurons for a Behaving Robot

R. L. B. French and R. I. Damper
Image, Speech and Intelligent Systems Research Group,
Department of Electronics and Computer Science,
University of Southampton,
Southampton SO17 1BJ, UK.
(emails: {rlbf98r|rid}@ecs.soton.ac.uk)

Abstract

We describe the artificial evolution of 'nervous systems' for the ARBIB robot, which control the way it interacts with its environment. The present work differs from earlier attempts to evolve robot controllers by use of a biologically-inspired heterogeneous network of spiking neurons. An obstacle-avoidance task is defined so as to provide an appropriate fitness function for the evolutionary process, and ten separate runs were undertaken in a simulated environment. Evolved nervous systems from 7 of the 10 runs led to an 'emergent' wall-following behaviour. Two of the seven, both having just 7 neurons, are examined and described. They are considerably simpler than our earlier, manually-designed solutions which had some 30-50 neurons, although the latter were additionally capable of ontogenetic (during lifetime) learning. One of the two example nervous system promotes phototaxis as well as wall-following. These two evolved systems are tested on a real Khepera robot, and behave entirely as expected from the simulations.

1 Introduction

Recently, there has been considerable attention paid to evolving controllers for autonomous robots. Evolutionary computation has contributed to robot development in several ways, with genetic algorithms (e.g., Davidor 1991) and genetic programming (e.g., Koza 1992) dominant in the field. Applications have been found in learning classifier systems (Dorigo 1995; Krebs and Bossel 1997; Smith and Cribbs 1997; Bertin and van de Grind 1998), navigation (Floreano and Mondada 1996; Chongstitvatana 1999; Nearchou 1999), obstacle avoidance (Ojala 1998), neural network design (Husbands, Cliff, Thompson, Jakobi, and Harvey 1997; Michel 1997), artificial life (Sims 1994a; 1994b), development of robot body shape (Funes and Pollack 1998) and the automated design and manufacture of robots (Lipson and Pollack 2000). The present work is aimed at evolving a 'nervous system' for the ARBIB autonomous robot. What distinguishes this from previous work is our use of a network of spiking neurons. To our knowledge, this is the first practical attempt to evolve a spiking nervous system artificially.

ARBIB is an Autonomous Robot Based on Inspirations from Biology which learns via low-level neural mechanisms of habituation, sensitisation and classical conditioning (Damper and Scutt 1998; Damper, French, and Scutt 2000). In engineering terms, it consists of a mobile platform (a Khepera miniature robot in this work) controlled by a neural network simulator called Hi-NOON which works at the basic level of nerve cell membrane potential, i.e., it is capable of simulating spiking neurons. This requires some justification. One might reasonably ask what advantages this offers, i.e., what can a simulation based on spiking neurons achieve that cannot be achieved using more gross parallel distributed processing (PDP) type model neurons? This is currently a vexed question in computational neuroscience. It is likely that spikes evolved mainly as a means of regenerative signaling along the relatively long propagation paths of animal nervous systems (Levitan and Kaczmarek 1997). Because of active regenerative processes, the action potential is propagated without loss of amplitude; hence, no information is carried by signal amplitude. Accordingly, time of firing plays a central role in neural coding, and this strongly suggests that a circuit of spiking neurons ought to be better able to handle temporal processing than PDP nets. Clearly, then, detailed timing information for individual spikes, and relative timing between spikes, offers an additional dimension to the neural code (Damper, French, and Scutt, forthcoming). There is suggestive evidence that this sort of information is indeed important in biology. Citing Rieke et al. (1997, p. 279):

> "... under many conditions, behavioural decisions are made with of order one spike per cell, ... individual spikes can convey several bits of information about incoming sensory stimuli ... precise discriminations could ... be based on the occurrence of individual spikes ..."

As previously implemented, ARBIB's nervous system was hand-coded. Hence, its architecture was fixed by the imagination and prejudices of its programmer. Neuroscience has not yet progressed to the state where we understand the relationships between nervous system structure and intelli-

gent behaviour. So it seems likely that manual design will limit the potential for the nervous system to scale-up and arguably hampers progress towards complex and intelligent behaviours. A possible solution to this problem, which we explore here, is to construct ARBIB's nervous system using the paradigm of evolutionary computation. Since our present concern is an *evolved* solution, we have disabled the mechanisms of ontogenetic (lifetime) learning by preventing the evolutionary process from constructing synapse-on-synapse connections which underlie the neural learning model of ARBIB (Damper, French, and Scutt 2000). This gives a clearer view of the effects of evolution and allows faster evaluation of candidate individuals.

The remainder of this paper is structured as follows. Section 2 briefly describes the way a nervous system is represented in Hi-NOON and the consequences this has for development through evolution. We show how simulated robot activity is used to evaluate a candidate nervous system, as well as describing the evolutionary processes of selection, recombination and mutation. Section 3 examines the structure and function of a typical nervous system evolved to promote obstacle avoidance in a simulated environment, but which also led to 'emergent' wall-following behaviour. Section 4 reports tests of this evolved network with a Khepera robot in the real world. Section 5 describes a second evolved nervous system, together with its atypical phototaxis (light seeking) ability. Section 6 concludes with suggestions for future development of this work.

2 Evolving a Nervous System

Hi-NOON is a simulator for a Hierarchical Network of Object-Oriented Neurons (Scutt and Damper 1991; French, Damper, and Scutt 2000; Damper, French, and Scutt, forthcoming). As the name suggests, synapses, neurons and network are represented as dynamically-created objects (equipped with their own data members and member functions) within an object-oriented hierarchy. Substantial machinery is already in place for executing a nervous system in Hi-NOON, and it is advantageous to reuse as much of this as possible during evolution. Hence, candidate nervous systems are represented and manipulated during selection, recombination and mutation using the existing data structures. The approach is in marked contrast to that adopted by, for example, Michel (1997) where a separate growth phase precedes nervous system execution. Direct manipulation of data structures avoids a (possibly) protracted growth phase.

In this work, we repeated the evolutionary process 10 times. Each run was for a population of 100 individuals (Hi-NOON nervous systems) over 100 generations.

2.1 Initial population

Initially, the evolutionary process must be seeded with a population of randomly-generated individuals. How are we to do this? The individuals in any new generation (including the initial one) must have a high likelihood of being structurally-viable nervous systems: Thus, it is advantageous to constrain (at least partially) the genetic operators to help achieve this. One way to do this is to ensure that newly-generated neurons of a particular type (e.g., sensory neuron) can only connect to other neurons of a particular type (e.g., interneuron). To maintain diversity, the constraints are sometimes applied and sometimes not, according to a random choice.

Hence, we use the following functions:

`add_neuron`, `delete_neuron`, `add_synapse` and `delete_synapse`—simply create and destroy neurons and synapses.

`add_sensor_to`—creates a sensory neuron at a random sensor position on ARBIB, and connects it to a randomly selected motor or interneuron postsynaptic target.

`add_turn`—creates an interneuron that carries an identification tag as a 'reflex' interneuron which then attempts to connect postsynaptically with motor neurons connected to the robot's differential drive.

`add_reflex`—creates an interneuron and connects it with a randomly-selected postsynaptic reflex interneuron.

`add_interneuron`—creates an interneuron and connects it to a randomly-selected motor, interneuron or reflex postsynaptic target.

`add_forward`—creates an interneuron and connects it to available forward motor postsynaptic targets.

These functions are used both to generate the initial population and in the creation of new generations during the subsequent evolutionary process itself, as illustrated in Figure 1.

2.2 Recombination and mutation

Recombination is implemented as single-point crossover. Two parents are chosen together with a common neuron (a neuron with a numerical identifier, the 'neuron number', common to both parent nervous systems) which acts as a crossover point. Two children are instantiated who then undergo pruning of any synapses that no longer have valid postsynaptic targets. In the interests of computational efficiency, this method has been chosen in preference to any attempt at analysing the two parents looking for an ideal crossover point. This operation is similar in function to the crossover used by Sims (1994b). Here, however, we only use one crossover point whereas Sims uses one or more. Also, he reassigns synapses that no longer have a postsynaptic target and we do not.

Mutation operates by adding or deleting individual neurons and synapses. A random choice is made as to whether to apply the functions listed in Section 2.1 or not. Because the Hi-NOON data structures are dynamic, and because we manipulate the nervous system representation directly during the evolutionary process, there is no limit on the size of genotype in this application. Genetic algorithms usually employ a fixed-length genotype (Koza 1992, p. 18): Hence, we see this

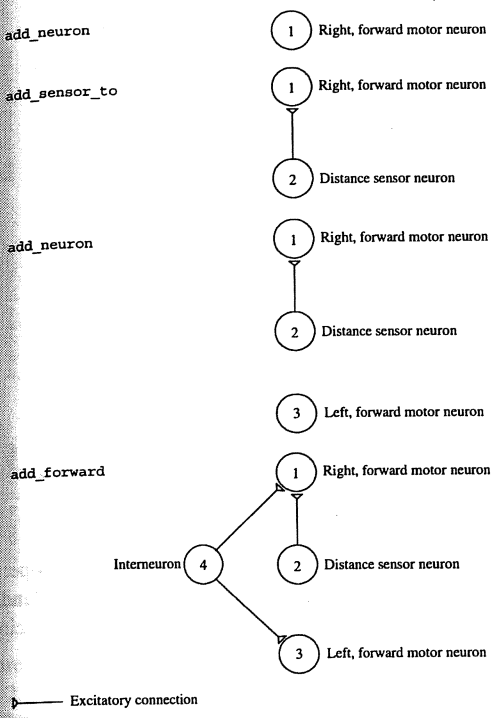

Figure 1: Illustration of some functions used to construct a nervous system (see text for details.)

development as being closer to the paradigm of genetic programming than genetic algorithms, with Hi-NOON viewed as a virtual machine.

2.3 Evaluation and fitness

Once a nervous system has been constructed, it must be evaluated either on the robot platform or in a simulation. Here, we perform evaluation in simulation for convenience. Essentially, each of the robot's sensors and motors is associated with an accumulator, which sums activity in that device over a fixed number of simulation time-steps, i.e., they are weighted averages. The accumulated values are then scaled, summed and divided by the number of time-steps. The scaling coefficients are predefined to reflect our intuitions about what obstacle avoidance involves. Hence, bump sensors have a negative coefficient but forward motor drives have a positive coefficient. In so doing, we are conscious of the observation of Zaera, Cliff, and Bruten (1996) that: "... formulating an effective fitness evaluation function for use in evolving controllers can be at least as difficult as hand-crafting an effective controller design." In our particular situation, however, an effective fitness function suggests itself rather easily.

Proceeding as above, we obtain an evaluation of the robot's performance per-clock-tick of nervous system execution, according to the following detailed equations:

$$
\begin{aligned}
S_{\mathrm{IR}} &= \sum_{i=0}^{7} k_i R_i \\
S_{\mathrm{light}} &= \sum_{i=0}^{7} k_{i+8} L_i \\
M_{\mathrm{l}} &= k_{16} M_{\mathrm{lf}} + k_{17} M_{\mathrm{lr}} \\
M_{\mathrm{r}} &= k_{18} M_{\mathrm{rf}} + k_{19} M_{\mathrm{rr}} \\
a_j &= \frac{S_{\mathrm{IR}} + S_{\mathrm{light}} + M_{\mathrm{l}} + M_{\mathrm{R}}}{T} \qquad \forall j
\end{aligned}
$$

Here, S_{IR} and S_{light} are the accumulated sums for the infrared (IR) range and light sensors, respectively; the range of the summation counter i is [0,7] because Khepera has 8 combined IR/light sensors; k_i denotes the ith coefficient; R_i and L_i are the ith infrared range and light activities, respectively; M_{l} and M_{r} are motor activities on the left and right sides, respectively; M_{lf} and M_{lr} are forward and reverse activities for the left motor; similarly M_{rf} and M_{rr} are forward and reverse activities for the right motor; T is the number of simulation time-steps; and a_j is the scaled accumulator value for the jth individual of the population.

The selection operator used here is the stochastic universal sampling (SUS) method (Baker 1987; Whitley 1993) with elitism. SUS was chosen because it is computationally efficient and maintains diversity among the population. However, some of the a_j's evaluated as above may be negative for particularly unfit individuals, whereas the SUS scheme requires sampling along a positive number axis divided up according to individual fitnesses. Hence, we determine fitness F_j for the jth individual by increasing the a_j values by the absolute minimum value of all the a_j's according to:

$$F_j = a_j + |\min[a_1, \ldots, a_P]|$$

where P is the size of the population. This results in a set of non-negative values to which SUS can be applied.

3 Typical (Simulated) Evolved Behaviour

In this section, we examine a typical result obtained by applying the evolutionary paradigm to nervous system development for the Khepera instantiation of ARBIB in a simulated environment (Michel 1996). The primary task for the robot was obstacle avoidance. The evolved nervous system was biased to produce this behaviour by the choice of accumulator coefficients, as described in the previous section.

Figure 2 shows a typical nervous system found in one of the 10 runs. It has several redundant components that perform no useful function here: They are 'leftovers' from the evolutionary process. Had evolution continued longer, it is possible that these might eventually have played a part in robot behaviour, i.e., they could serve as preadaptations for some as yet undiscovered functionality.

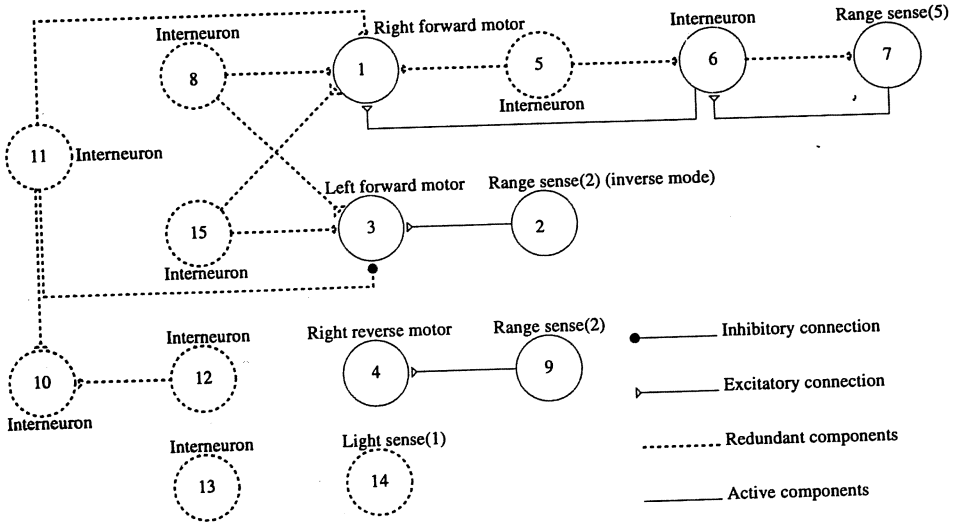

Figure 2: Typical nervous system evolved for obstacle avoidance. Neuron 2 is depolarised by an IR distance sensor that operates in inverse mode. Numbers in brackets after sensory neurons denote position on Khepera. Some components are redundant: They have no sensory input or are disconnected from postsynaptic targets. On removal of redundant components, the circuit divides into three separate subassemblies.

As can be seen in the figure, the nervous system has organised itself into three separate parts:

1. neurons 1, 6 and 7 form a subassembly that fires the right forward motor when a wall is detected by distance sensor 5 (a wall on the robot's right hand side);
2. neurons 2 and 3 fire the left forward motor in the absence of an obstacle in front of the robot;
3. neurons 4 and 9 fire the right reverse motor in the presence of an obstacle in front of the robot.

Note that only the IR range sensors are in use. Although a light sensory neuron has been created, it has not been connected to any other part of the nervous system. The first subassembly above is only one step away from forming an oscillatory neural circuit—were it not for the fact that neuron 7 is a sensory cell and so its membrane potential is heavily influenced by the environment.

Figure 3 shows the path taken by ARBIB (with the nervous system shown in Figure 2) in a test environment. It is abundantly clear that it shows robust wall-following behaviour. Interestingly, 7 out of the 10 separate evaluations resulted in such behaviour. The remaining 3 runs generated nervous systems that predisposed the (simulated) robot either to explore a limited part of the environment by traversing a small circular path, or to remain stationary on encountering a wall.

Whether or not this wall-following is 'emergent' depends upon one's interpretation of this rather loose term. For instance, Steels (1994) has described wall-following behaviour in a robot as 'emergent' since it was not explicitly built into the controller. On the other hand, Ronald, Sipper, and Capcarrere (1999) discount this use of the term because tendency

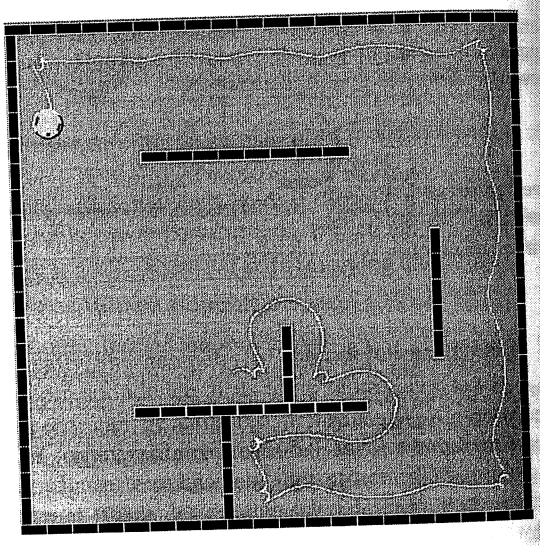

Figure 3: A screenshot of ARBIB exploring a simulated environment using the evolved nervous system in Figure 2. Wall-following behaviour has emerged.

to follow a wall can be predicted from knowledge of two underlying simpler behaviours—obstacle avoidance and forward motion in the direction of a wall. To these authors, an element of surprise is necessary for a behaviour to qualify as emergent. On the other hand, Damper (2000) points out that surprise is observer-dependent, and no scientific test should be relative to the degree of knowledge or ignorance of the observer. Indeed, the behaviour seen here surprised one of the authors but not the other! So rather than labour the point of 'emergence', we simply emphasise that wall-following was

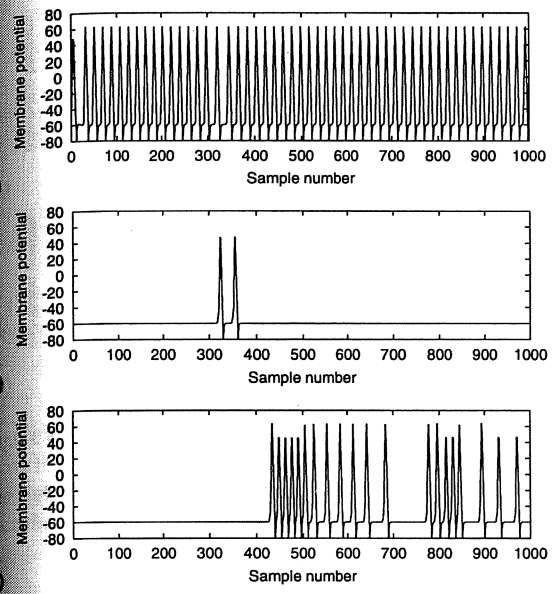

Figure 4: Typical spiking activity in the nervous system of Fig. 2. (a) Activity in the left forward motor neuron. Initially, nothing is detected in front of ARBIB so the left forward motor fires between samples [0, 300]. (b) Activity in the right reverse motor neuron occurs just after sample 300. Because ARBIB has turned to face a wall, neuron 2 briefly stops firing, causing neuron 9 to fire. (c) Activity in the right forward motor neuron. Together with (a), this shows that ARBIB is moving forward during the time between [450, 700].

not programmed into the controller: only a fitness function for obstacle avoidance was. This, after all, is our motivation for using the evolutionary paradigm—so that we are not constrained by the foresight of a human designer.

To help visualise what is going on as ARBIB explores its world, membrane potentials from all neurons are logged to disk for inspection. A total of 10019 sample points was collected for each neuron, but only a small subset will be displayed here.

Initially, with ARBIB facing towards the top of the environment in Figure 3, only range sensor neuron 2 (see Fig. 2) is firing because this neuron operates in inverse mode, and nothing is detected. Hence, it fires the left forward motor, as shown in Figure 4(a) for samples [0, 300]. This activity orients ARBIB to the right because the right motor is inactive. Figure 4(b) shows that activity in the right reverse motor is limited to the period just after sample 300, because ARBIB has turned to face a wall, briefly stopping neuron 2 from firing and causing sensory neuron 9 to fire. Hence, the robot turns right. Once the wall is out of range of sensory neuron 9, neuron 2 fires again, resuming forward motion in the left motor. As neuron 7 has not yet seen the wall, the right forward motor is at rest and ARBIB turns right using the left forward motor until neuron 7 first sees the wall, fires, and the robot moves forward (Fig. 4(a) and (c), samples [450, 700]). So, although each of the three subassemblies is itself very simple, the overall behaviour can be rather complex.

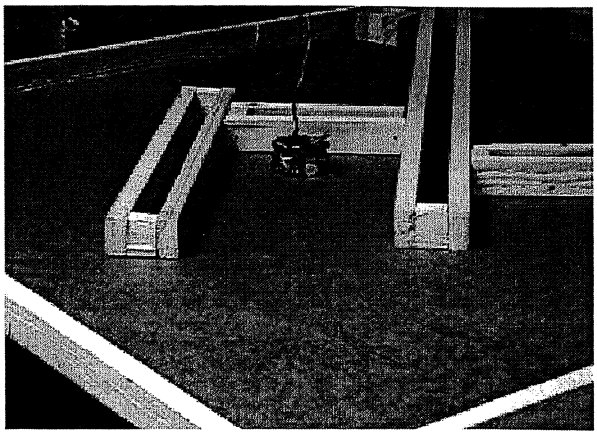

Figure 5: ARBIB negotiating a test environment using the evolved spiking nervous system of Figure 2.

How different is this 'nervous system' from one that we might have designed manually? First and foremost, it is considerably simpler. The evolutionary solution has just 7 neurons, whereas our earlier, manually-designed solutions had some 30-50 neurons. Our manual designs have used central pattern generator circuits (Selverston 1988; Kleinfield and Sompolinsky 1989) to give ARBIB a drive to explore its environment, together with a hardwired obstacle-avoidance reflex which was modified by learning (classical conditioning). Here, however, there is no ontogenetic learning so, by focusing crudely on neuron count, we are not strictly comparing like with like. Also, the division of the evolved solution into three separate subassemblies was interesting. All our previous designs featured a single, complete neural circuit.

4 Real-World Evaluation

Results from the simulations described in Section 3 were verified using a real Khepera robot. No changes to the evolved nervous system of Fig. 2 were made.

A Khepera robot in mode 3 (controlled by serial link protocol at 38400 Baud) was placed into a test environment (Figure 5), facing away from the nearest wall. The Khepera graphical user interface (GUI) of Michel (1996) was used as the front-end for Hi-NOON.

After selection of the real Khepera (through the GUI), ARBIB started to search for a wall. If nothing was in view, it rotated until a wall was seen. It then approached the wall, oriented itself parallel to it, and moved forwards. On encountering a corner, it adjusted its position and orientation and continued along the new wall. On one occasion, a new object (a black coffee jar) was introduced into the environment and directly in ARBIB's path. The robot manoeuvred around the object until its top circuit board hit the jar, whereupon it got stuck. At this point, the first author intervened and unjammed the robot,

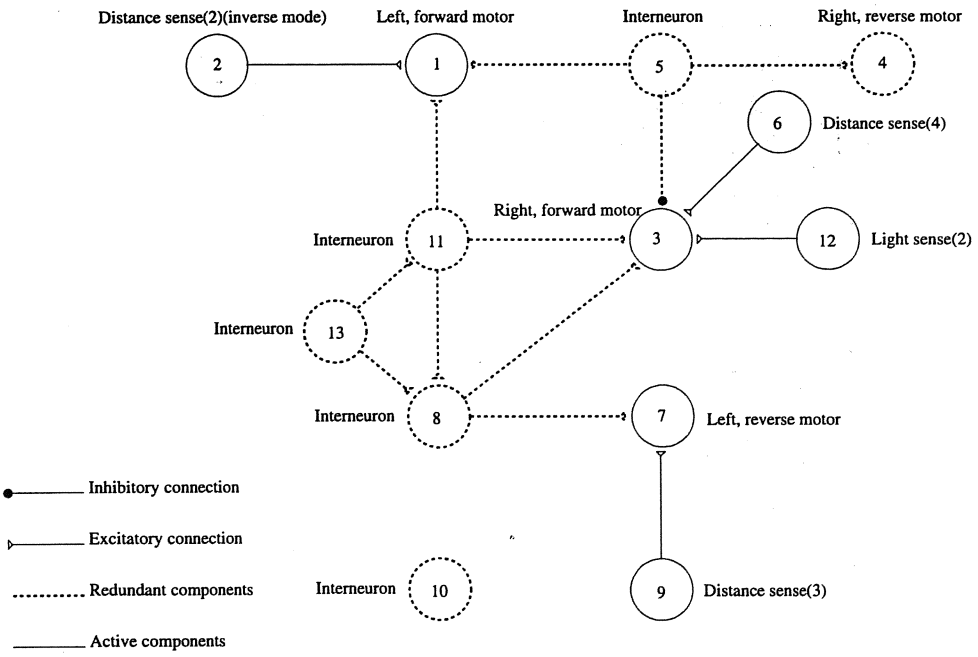

Figure 6: A nervous system that leads to phototaxis instead of wall following when in the presence of a light source.

which then negotiated around the jar, found a wall and continued its journey around the periphery of the environment.

Hence, a system of spiking neurons evolved for a simulated robot transfers very well to the real world. This robustness probably relies largely on the sensory and motor noise which is deliberately introduced as a feature of the Khepera simulator (cf. Chongstitvatana 1999).

5 An Atypical Result—Phototaxis

One of the evolved nervous systems produced entirely atypical behaviour when evaluated (both in simulation and in the real robot). This nervous system, shown in Figure 6, not only gave rise to wall following but to phototaxis also. Like the circuit of Figure 2, it has 7 neurons grouped into 3 separate subassemblies. However, unlike the earlier nervous system, it has developed a light sensory neuron presynaptic to a motor neuron (which is also postsynaptic to a distance sensory neuron). This additional sensory modality makes ARBIB head towards a nearby light source (phototaxis) in preference to wall-following, and so in a sense it has a behavioural hierarchy.

Figure 7 depicts simulated behaviour of ARBIB equipped with this nervous system. Initially positioned at the bottom right of the virtual environment, away from the side and facing towards the top of the frame, the only sensory neuron that fires will be neuron 2 because it is configured for inverse mode (i.e., firing inversely proportional to sensed distance). Hence, the left forward motor fires and turns ARBIB to the right until sensory neuron 6 detects the wall, fires the right forward motor, and the robot follows the wall. As ARBIB approaches the right hand side of the environment, sensory neuron 2 stops

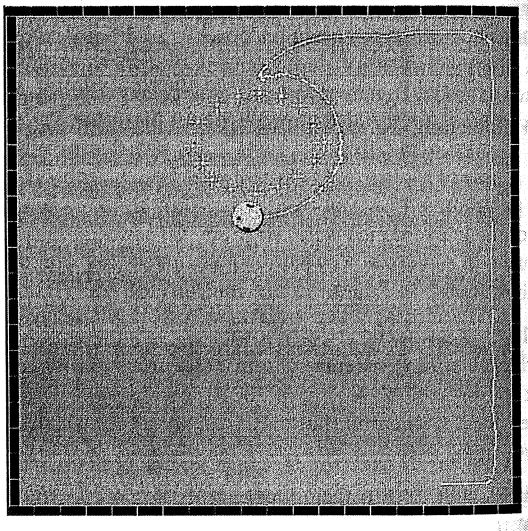

Figure 7: When controlling a simulated robot, the nervous system of Figure 6 leads to atypical phototaxis towards a cluster of (simulated) lights, in addition to wall following.

firing and so does the left forward motor. A wall is now detected by sensory neuron 9 causing the left reverse motor to fire and the robot turns left, ready to follow the right hand side wall of the environment. Once sensory neuron 2 starts firing again, ARBIB moves forward. Eventually it reaches the top of the wall (now at the top right hand corner of the frame), sensory neuron 9 fires, and ARBIB turns left to follow the new wall. However, as it approaches the light source (on its left), sensory neuron 12 fires, and coupled with activity in sensory

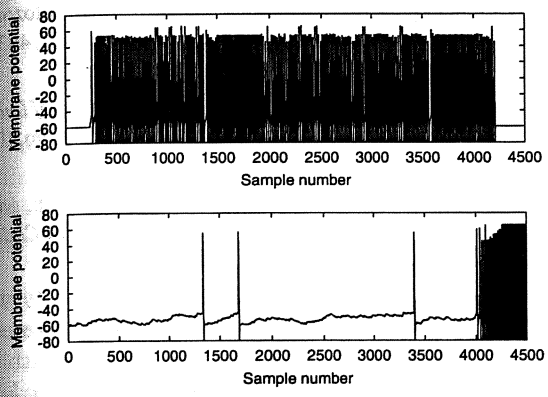

Figure 8: (a) Activity in distance sensory neuron 6, and (b) activity in light sensory neuron 12. When both are depolarising the right forward motor, the latter fires and ARBIB turns to face the light source.

neuron 6 (see Fig. 8(a) and (b) around sample 4000) causes a bias in the robot's path—it starts to arc towards the light because of increased activity in the right forward motor (the motor is now excited by sensory neurons 6 and 12). Sensory neuron 6 eventually stops firing when it is too far away from the wall to be depolarised. However, the right forward motor is still excited by sensory neuron 12 and so ARBIB moves forward towards the lamps. On colliding with the lamps, sensory neuron 9 fires and the robot turns left. ARBIB now starts to trace around the cluster of (simulated) light sources.

Following the test in a simulated environment, this nervous system was tested with a real Khepera. The real robot showed a basic wall-following behaviour but consistently showed phototaxis by abandoning this and heading towards a lamp placed in its vicinity.

6 Conclusion and Future Work

We have described an application of evolutionary computation to the construction of a spiking-neuron controller for the ARBIB autonomous robot. To our knowledge, this is the first attempt to evolve a spiking nervous system for a behaving robot. The normal mechanisms of ontogenetic learning in ARBIB were disabled for this study. Ten runs of the evolutionary process were conducted with evaluation of fitness in a simulated environment. Evaluation was biased to favour obstacle avoidance. Seven out of ten runs resulted in robust wall-following behaviour. Two nervous systems were selected from these seven runs, and their operation examined. Both had just 7 neurons—considerably simpler than any of our previously-designed neural controllers for ARBIB, although these more complex circuits featured ontogenetic learning. One of the two selected nervous systems was found to promote phototaxis (light seeking). It is unclear whether wall-following and phototaxis count as 'emergent behaviour' or not, but our approach does seem to offer at least the chance of producing novel, useful behaviours. In both cases, a real Khepera was tested with the evolved controllers and behaved entirely as expected from the simulations.

Future work will reintroduce the mechanisms of ontogenetic learning based on synaptic plasticity, to understand how this interacts with evolution. There is, of course, good reason to expect benefits from this interaction, as renewed interest in the Baldwin effect (Baldwin 1896; Richards 1987) is making clear. The basic insight is that the ability of an individual to learn can promote fitness so as to improve chances of selection, and can be genetically transmitted without the learned information itself being transmitted in Lamarckian fashion. The advantages that this confers have been confirmed by several neural network simulations (Hinton and Nowlan 1987; Gruau and Whitley 1993; Nolfi, Parisi, and Elman 1994; Nolfi 1999).

References

Baker, J. E. (1987). Reducing bias and inefficiency in the selection algorithm. In J. J. Grenfenstett (Ed.), *Genetic Algorithms and their Applications: Proceedings of the 2nd International Conference on Genetic Algorithms*, Hillsdale, NJ, pp. 14–21. Lawrence Erlbaum Associates.

Baldwin, J. M. (1896). A new factor in evolution. *American Naturalist 30*, 441–451.

Bertin, R. J. V. and W. A. van de Grind (1998). Phototaxic foraging of the Archaepaddler, a hypothetical deep-sea species. *Artificial Life 4*(2), 157–181.

Chongstitvatana, P. (1999). Using perturbation to improve robustness of solutions generated by genetic programming for robot learning. *Journal of Circuits, Systems, and Computers 9*(1–2), 133–143.

Damper, R. I. (2000). Emergence and levels of abstraction. *International Journal of Systems Science 31*(7), 811–818.

Damper, R. I., R. L. B. French, and T. W. Scutt (2000). ARBIB: an autonomous robot based on inspirations from biology. *Robotics and Autonomous Systems 31*(4), 247–274.

Damper, R. I., R. L. B. French, and T. W. Scutt (forthcoming). The Hi-NOON neural simulator and its applications. *The Microelectronics Journal*, in press.

Damper, R. I. and T. W. Scutt (1998). Biologically-based learning in the ARBIB autonomous robot. In *Proceedings of IEEE International Symposia on Intelligence and Systems*, Washington DC, pp. 49–56.

Davidor, Y. (1991). *Genetic Algorithms and Robotics*. Singapore: World Scientific Press.

Dorigo, M. (1995). Alecsys and the autonomouse: Learning to control a real robot by distributed classifier systems. *Machine Learning 19*(3), 209–240.

Floreano, D. and F. Mondada (1996). Evolution of homing navigation in a real mobile robot. *IEEE Transactions*

on Systems, Man, and Cybernetics—Part B: Cybernetics 26(3), 396–407.

French, R. L. B., R. I. Damper, and T. W. Scutt (2000). The Hi-NOON neural simulator and its applications to animal, animat and humanoid studies. *First IEEE-RAS International Conference on Humanoid Robots*, Boston, MA. (No pagination; Proceedings on CD-ROM).

Funes, P. and J. B. Pollack (1998). Evolutionary body building: Adaptive physical designs for robots. *Artificial Life 4*(4), 337–357.

Gruau, F. and D. Whitley (1993). Adding learning to the cellular development of neural networks: Evolution and the Baldwin effect. *Evolutionary Computation 1*(3), 213–233.

Hinton, G. E. and S. Nowlan (1987). How learning can guide evolution. *Complex Systems 1*(3), 495–502.

Husbands, P., D. Cliff, A. Thompson, N. Jakobi, and I. Harvey (1997). Evolutionary robotics: The Sussex approach. *Robotics and Autonomous Systems 20*(2–4), 205–224.

Kleinfield, D. and H. Sompolinsky (1989). Associative neural networks for central pattern generators. In C. Koch and I. Segev (Eds.), *Methods in Neuronal Modeling: From Synapses to Networks*, pp. 195–246. Cambridge, MA: MIT Press.

Koza, J. R. (1992). *Genetic Programming: On the Programming of Computers by Means of Natural Selection*. Cambridge, MA: MIT Press/Bradford Books.

Krebs, F. and H. Bossel (1997). Emergent value orientation in self-organization of an animat. *Ecological Modelling 96*(1–3), 143–164.

Levitan, I. B. and L. K. Kaczmarek (1997). *The Neuron: Cell and Molecular Biology*. New York, NY: Oxford University Press.

Lipson, H. and J. B. Pollack (2000). Automatic design and manufacture of robotic lifeforms. *Nature 406*(6799), 974–978.

Michel, O. (1996). Khepera Simulator version 2, User Manual, 1 March 1996. Downloadable from URL `http://diwww.epfl.ch/Khepera`.

Michel, O. (1997). Dynamical genomic network applied to artificial neurogenesis. *Control and Cybernetics 26*(3), 511–531.

Nearchou, A. C. (1999). A genetic navigation algorithm for autonomous mobile robots. *Cybernetics and Systems 30*(7), 629–661.

Nolfi, S. (1999). How learning and evolution interact: The case of a learning task which differs from the evolutionary task. *Adaptive Behavior 7*(2), 231–236.

Nolfi, S., D. Parisi, and J. L. Elman (1994). Learning and evolution in neural networks. *Adaptive Behavior 3*(1), 5–28.

Ojala, A. (1998). Adaptive behavior with protozoa-inspired dynamics. *Biological Cybernetics 79*(5), 403–411.

Richards, R. J. (1987). *Darwin and the Emergence of Evolutionary Theories of Mind and Behavior*. Chicago, IL: University of Chicago Press.

Rieke, F., D. Warland, R. de Ruyter van Steveninck, and W. Bialek (1997). *Spikes: Exploring the Neural Code*. Cambridge, MA: Bradford Books/MIT Press.

Ronald, E. M. A., M. Sipper, and M. S. Capcarrere (1999). Design, observation, surprise! A test of emergence. *Artificial Life 5*(3), 225–239.

Scutt, T. W. and R. I. Damper (1991). Computational modelling of learning and behaviour in small neuronal systems. In *Proceedings of International Joint Conference on Neural Networks*, Singapore, pp. 430–435.

Selverston, A. I. (1988). A consideration of invertebrate pattern generators as computational databases. *Neural Networks 1*(2), 109–117.

Sims, K. (1994a). Evolving 3D morphology and behavior by competition. In R. A. Brooks and P. Maes (Eds.), *Artificial Life IV*, pp. 28–39. Cambridge, MA: MIT Press.

Sims, K. (1994b). Evolving virtual creatures. In A. Glassner (Ed.), *SIGGRAPH 94 Conference Proceedings*, pp. 15–22. New York, NY: ACM Press.

Smith, R. E. and H. B. Cribbs (1997). Combined biological paradigms: A neural genetics-based autonomous systems strategy. *Robotics and Autonomous Systems 22*(1), 65–74.

Steels, L. (1994). The artificial life roots of artificial intelligence. *Artificial Life Journal 1*(1), 89–125.

Whitley, D. (1993). A genetic algorithm tutorial. *Statistics and Computing 2*(4), 65–85.

Zaera, N., D. Cliff, and J. Bruten (1996). (Not) evolving collective behaviours in synthetic fish. In P. Maes, M. Mataric, J.-A. Meyer, J. Pollack, and S. Wilson (Eds.), *From Animals to Animats 4: Proceedings of the 4th International Conference on Simulation of Adaptive Behavior (SAB96)*, pp. 635–644. Cambridge, MA: MIT Press/Bradford Books.

Proceedings of the Genetic and Evolutionary Computation Conference

Edited by

Lee Spector

Erik D. Goodman

Annie Wu

W. B. Langdon

Hans-Michael Voigt

Mitsuo Gen

Sandip Sen

Marco Dorigo

Shahram Pezeshk

Max H. Garzon

Edmund Burke

July 7–11, 2001
San Francisco,
California