CSRP 497
Running Across the Reality Gap: Octopod Locomotion Evolved in a Minimal Simulation*

Nick Jakobi

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

Abstract. This paper describes experiments in which neural network control architectures were evolved in minimal simulation for an octopod robot. The robot is around 30cm long and has 4 infra red sensors that point ahead and to the side, various bumpers and whiskers, and ten ambient light sensors positioned strategically around the body. Each of the robot's eight legs is controlled by two servo motors, one for movement in the horizontal plane, and one for movement in the vertical plane, which means that the robots motors have a total of sixteen degrees of freedom. The aim of the experiments was to evolve neural network control architectures that would allow the robot to wander around its environment avoiding objects using its infra-red sensors and backing away from objects that it hits with its bumpers. This is a hard behaviour to evolve when one considers that in order to achieve any sort of coherent movement the controller has to control not just one or two motors in a coordinated fashion but sixteen. Moreover it is an extremely difficult set-up to simulate using traditional techniques since the physical outcome of sixteen motor movements is rarely predictable in all but the simplest cases. The evolution of this behaviour in a minimal simulation, with perfect transference to reality, therefore, provides essential evidence that complex motor behaviours can be evolved in simulations built according to the theory and methodology of minimal simulations.

1 Introduction

Evolutionary Robotics is not magic and as several authors have pointed out [2, 6, 12, 14], there are many big questions that need answers if Evolutionary Robotics is to progress beyond the proof of concept stage. One of the most urgent of these (in that if it is not answered, Evolutionary Robotics is simulation. Space limitations do not allow a full explication of the theory and methodology of minimal simulations. However, the next section gives a broad brush sketch of the main ideas, which, together with the details of the minimal simulation used for the experiments reported in this paper, should give the reader a good idea of how to build and use a minimal simulation.

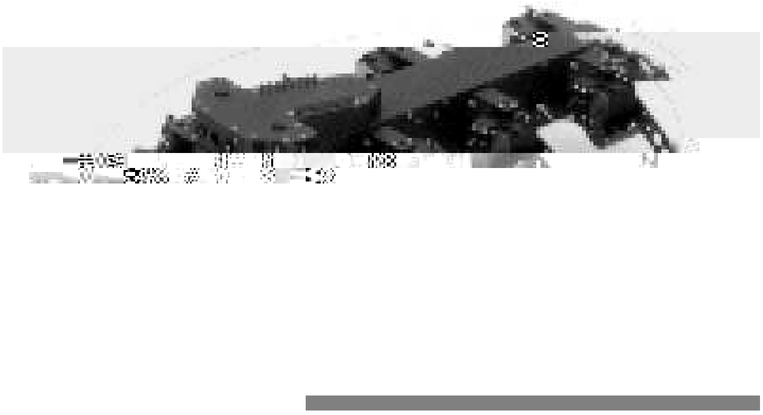


Fig. 1. The Octopod robot.

This paper describes experiments in which neural network control architectures were evolved for an octopod robot. The robot, shown in figure 1 is around 30cm long and has 4 infra red sensors that point ahead and to the side, various bumpers and whiskers, and ten ambient light sensors positioned strategically around the body. Each of the robot's eight legs is controlled by two servo motors, one for movement in the horizontal plane, and one for movement in the vertical plane, which means that the robots motors have a total of sixteen degrees of freedom.

The aim of the experiments was to evolve neural network control architectures that would allow the robot to wander around its environment avoiding objects using its infra-red sensors and backing away from objects that it hits with its bumpers. This is a hard behaviour to evolve when one considers that in order to achieve any sort of coherent movement the controller has to control not just one or two motors in a coordinated fashion but sixteen. Moreover it is an extremely difficult set-up to simulate using traditional techniques since the physical outcome of sixteen motor movements is rarely predictable in all but the simplest cases. The evolution of this behaviour in a minimal simulation, therefore, provides essential evidence that complex motor behaviours can be evolved in simulations built according to the theory and methodology put forward in [9, 8, 7]. See [3, 4, 11] for related work on evolving locomotion controllers for walking robots and for abstract computer models of insects.

The minimal simulation used to evolve controllers for the octopod is described in Section 3. The rest of the evolutionary machinery, including the neural networks, the encoding scheme, the genetic algorithm and genetic operators is described in section 4. Experimental results are put forward in section 5 and finally, in section 6, some comments are offered on the paper as a whole. But first, a brief description of the idea of minimal simulations.

around its environment in an acceptable manner, its legs do *not* clash and its belly does *not* drag along the ground and its legs do *not* pull in different directions. The minimal simulation described below takes full advantage of this fact. The dynamics of the simulated robot match the dynamics of the how the body as a whole moves in response. These include the way in which controller output affects how the legs move, and the way in which the movement of the legs affects the movement of the body as a whole.

Build a model of the way in which the members of the base set interact with each other and react to controller output (when the robot is performing the behaviour).

The overall movement of the robot was described by two variables: one for the speed of the left-hand side of the robot and one for the speed of the right-hand side of the robot. Thus if both sides of the robot moved straight ahead at the same speed then the overall movement of the robot was deemed to be straight ahead, if they moved in different directions but with equal velocity then the robot was deemed to be turning on the spot, and if both sides moved backwards at the same speed then the overall movement of the robot was deemed to be straight backwards.

To model the way in which the robot as a whole moved in response to controller output, therefore, necessitated a model of the way in which each leg responded to controller output, and the way in which the movement of each leg contributed to the overall movement of each *side* of the robot. However, because of the arguments put forwards in [9], it was not necessary to accurately model the

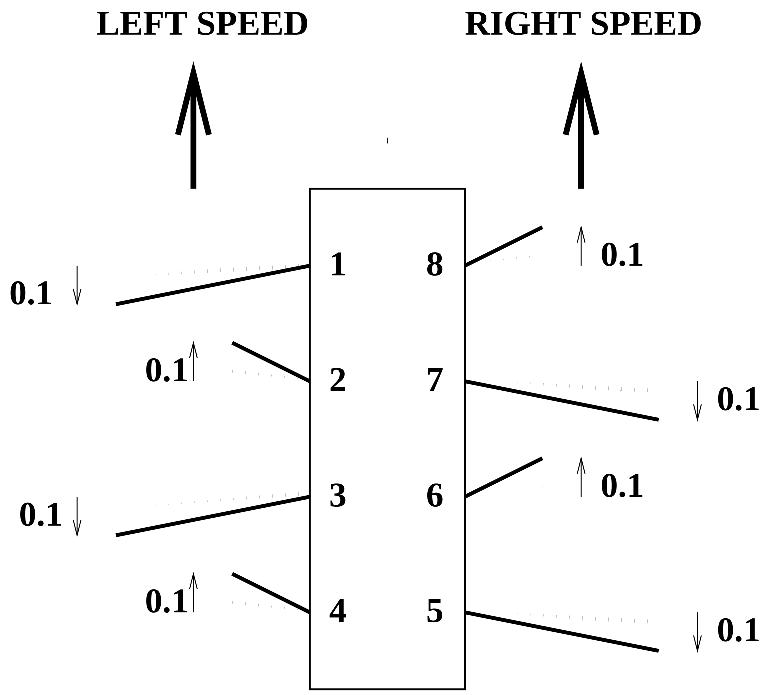
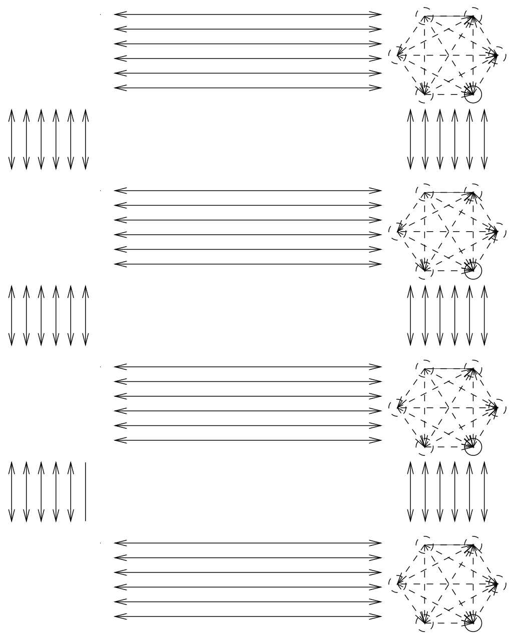

LEFT SPEED
RIGHT SPEED
1
2
3
4
5
6
7
8
0.1
0.1
0.1
0.1
0.1
0.1
0.1
0.1

Fig. 2.

Activation proportional to



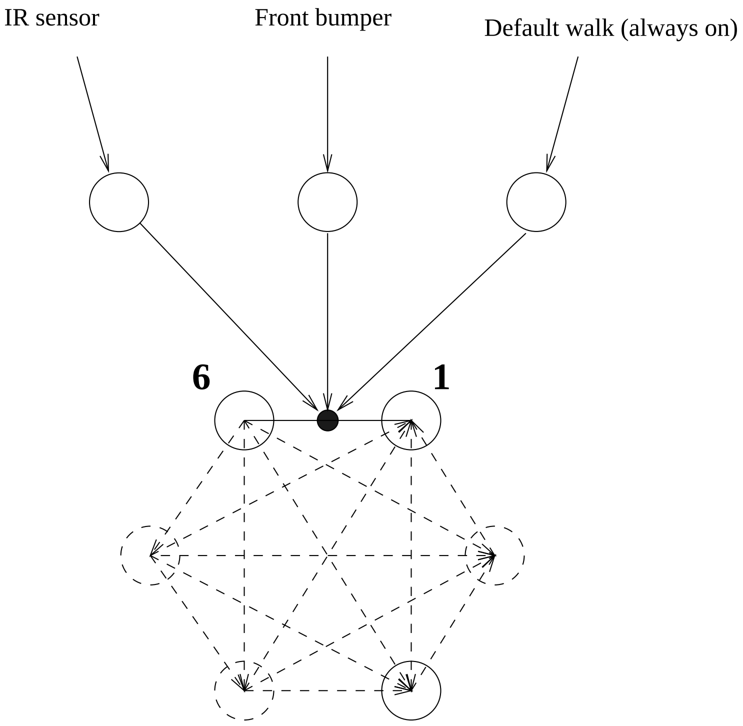
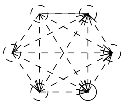
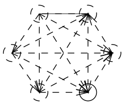
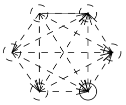
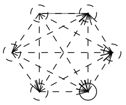

IR sensor
Front bumper
Default walk (always on)
6
1

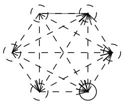
Fig. 5. This diagram shows how each connection between leg-controller neurons contains a synapse 'gate' that can be turned on or off by sensor neurons and the bias neuron. For the sake of diagrammatic simplicity only one connection is shown, whereas in reality every connection in every leg controller sub-network ($36 \times 8 = 288$

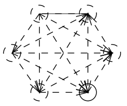












real number in the range 0 to 99, and this was mapped onto the relevant range during decoding. The parameters that were encoded and the ranges onto which they were mapped are as follows:

- 36 connection weights for the leg-controller sub-networks mapped onto the range ±16.
- 12 cross-body and along-body coupling connection weights mapped onto the range ±16.
- 36 infra-red sensor neuron to synapse connection weights mapped onto the range -6.5 to 25.5.
- 36 bumper sensor neuron to synapse connection weights mapped onto the range -6.5 to 25.5.
- 36 bias neuron to synapse connection weights mapped onto the range -6.5 to 25.5.
- 9 unit threshold constants mapped onto the range ±4: 6 for the leg-controller sub-network neurons, 1 for the infra-red sensor neurons, 1 for the bumper sensor neuron and 1 for the bias neuron.
- 9 unit time constants mapped onto the range 0.5 to 5.0: 6 for the leg-controller sub-network neurons, 1 for the infra-red sensor neurons, 1 for the bumper sensor neuron and 1 for the bias neuron.
- 3 input connection weights mapped onto the range ±16: 1 for the infra-red sensor neurons, 1 for the bumper sensor neuron and one for the bias neuron.

which makes a total of 177 parameters. Thus genotypes were strings of 177 numbers in the range 0 to 99.

Genetic algorithm and genetic operators

The genetic algorithm was an extremely simple generational model with tournament selection and elitism. After evaluating every member of the population, offspring genotypes were repeatedly produced until the next generation was full. To make a new offspring, two pairs of individuals were picked at random from the population and the fittest individuals from each pair (i.e. the winners of the tournaments) were chosen to act as parents. The offspring genotype was then formed from these two parents through a process of crossover and mutation: single point crossover was applied with a probability of 1, and every one of the 177 numbers that made up the offspring had a 0.02 chance of being mutated. A mutation involved changing the number in question by a random amount taken from a roughly normal distribution with a standard deviation of around 18. If the new value was greater than 99 or less that 0 then it was clipped to lie within this range.

5 Experimental results

After removing some initial bugs from the code[2], reliably fit controllers evolved on practically every run within around 3500 generations. This took around 14 hours to run on a Sun Ultra SPARC and simulated over 11 weeks worth of real world evolution. When downloaded onto the real octopod, reliably fit controllers made the robot walk around its environment in a satisfactory manner, turning this type and data of the required type is not available. The other form such a demonstration could take, and probably the most natural, is the evidence provided by video footage of the robot wandering around its environment. This cannot, however, be profitably presented as part of a text and pictures document; even if a sequence of stills taken at short and regular time intervals were displayed, this would not be all that informative as to how the legs of the robot moved in the real world unless there were an impractically large number of them.

In lieu of any method of demonstrating how the legs of the real robot moved as it wandered around its environment, the best we can do is to provide a demonstration of how the motor signal patterns to these legs change in response to each of the four sensory scenarios. Figure 7 offers such a demonstration for a typical reliably fit controller that evolved after 3200 generations. From top to bottom, the first eight traces provide a novel representation of the motor signals issued to each leg over the course of an average fitness trial, and the bottom two traces show the resultant velocities of the left and right side of the simulated robot respectively. The best way of explaining how to read the slightly bizarre looking motor traces is to describe how they were generated. At each iteration of the simulation, a short line representing the current motor signal was added to the right hand side of each motor signal trace. As can be seen from the figure, these lines were of various thicknesses and were always drawn from the horizontal centre line of the trace either up and to the left or down and to the left with various different gradients. The thickness of each line represented the vertical angle of the leg relative to the ground as specified by the motor signal in question: the thicker the line, the lower the leg, and the thinner the line the higher the leg. The gradient of the line represented the horizontal angle of the leg relative to the body as specified by the motor signal in question: the further up and to the left, the further forwards relative to the body, and the further down and to the left, the further backwards relative to the body. In this way, although they are perhaps harder to read than other less informative types of trace devised to convey similar information (see [1] for example), each of the motor signal traces of figure 7 represents *both*

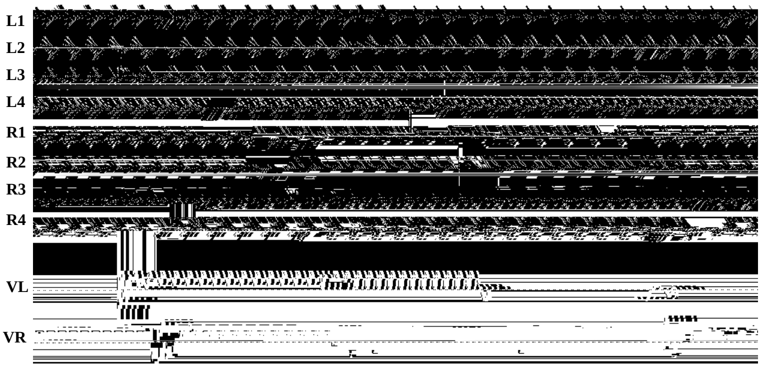


Fig. 7. Each leg controller consisted of six fully connected neurons. The activity of neuron 1 and neuron 2 controlled the horizontal and vertical leg angles respectively.

6 Comments

The minimal simulation used in this paper makes full use of the arguments put forward in [9] to evolve controllers for the octopod robot. Simply put, these arguments state that a minimal simulation need only model the real-world dynamics involved in successful behaviour and no others. This is because the only controllers that must cross the reality gap, if the simulation is to be a success, are precisely those that use these dynamics (i.e. perform the behaviour) and no others. For many robotics setups and behaviours this may not be of any use since the dynamics involved in successful behaviour may be neither obvious ahead of time nor qualitatively different to the rest of the dynamics of the system. For the experiments reported in this article, however, the dynamics of the octopod robot during successful walking and obstacle avoiding behaviour were both relatively easy to identify and *much* easier to model than the dynamics of the octopod robot as a whole. A minimal simulation that modelled these dynamics alone was therefore easy to construct and ran extremely fast when compared to the simulation that would result from attempting to model *all* of the dynamics of the octopod robot within its environment.

Acknowledgements

First, infinitely many thanks to Phil Husbands for editing this paper out of the thesis chapters I left him while I disappeared on the piste. What a fine man. Blame the title on him. Second, many thanks to Phil Husbands, Inman Harvey, Mike Wheeler, Giles Mayley, Joe Faith, Adam Bockrath, Seth Bullock, Jon Bird, Pete de Bourcier, Emmet Spier, Adrian Thompson, Suzy Levy and other members of COGS and the CCNR, past and present, for crucial discussions, debate and help. Third, a big salute to the Brighton beach 4:00am naked winter swimming club, I couldn't have done this without you. This work was supported by a COGS postgraduate bursary.

References

1. R.D. Beer and J.C. Gallagher. Evolving dynamic neural networks for adaptive behavior. *Adaptive Behavior*, 1:91–122, 1992.
2. R. Brooks. Artificial life and real robots. In F. J. Varela and P. Bourgine, editors, *Toward a Practice of Autonomous Systems: Proceedings of the first European Conference on Artificial Life*, pages 3–10, Cambridge, Massachusetts, 1992. MIT Press / Bradford Books.

3. F. Gruau. Automatic definition of modular neural networks. *Adaptive Behavior*, 3(2):151–184, 1995.
4. F. Gruau. Cellular encoding for interative evolutionary robotics. In P. Husbands and I. Harvey, editors, *Fourth European Conference on Artificial Life*, pages 368–377. MIT Press/Bradford Books, 1997.
5. I. Harvey and P. Husbands. Evolutionary robotics. In *Proceedings of IEE Colloquium on 'Genetic Algorithms for Control Systems Engineering', London 8 May 1992*, 1992.
6. P. Husbands and I. Harvey. Evolution versus design: Controlling autonomous robots. In *Integrating Perception, Planning and Action, Proceedings of 3rd Annual Conference on Artificial Intelligence, Simulation and Planning*, pages 139–146. IEEE Press, 1992.
7. N. Jakobi. Half-baked, ad-hoc and noisy: Minimal simulations for evolutionary robotics. In P. Husbands and I. Harvey, editors, *Fourth European Conference on Artificial Life*, pages 348–357. MIT Press/Bradford Books, 1997.
8. N. Jakobi. Evolutionary robotics and the radical envelope of noise hypothesis. *Adaptive Behavior*, 6(2):325–368, 1998.
9. N. Jakobi. *Minimal Simulations for Evolutionary Robotics*. PhD thesis, University of Sussex, 1998.
10. N. Jakobi, P. Husbands, and I. Harvey. Noise and the reality gap: The use of simulation in evolutionary robotics. In F. Moran, A. Moreno, J.J. Merelo, and P. Chacon, editors, *Advances in Artificial Life: Proc. 3rd European Conference on Artificial Life*, pages 704–720. Springer-Verlag, Lecture Notes in Artificial Intelligence 929, 1995.
11. J. Kodjabachian and J.-A. Meyer. Evolution and development of neural networks controlling locomotion, gradient following and obstacle avoidance in artificial insects. *IEEE Transactions on Neural Networks*, page (in press), 1998.
12. M.J. Mataric and D. Cliff. Challenges in evolving controllers for physical robots. *Robot and Autonomous Systems*, 19(1):67–83, 1996.
13. O. Miglino, H.H. Lund, and S. Nolfi. Evolving mobile robots in simulated and real environments. *Artifical Life*, 2(4), 1995.
14. S. Nolfi, D. Floreano, O. Miglino, and F. Mondada. How to evolve autonomous robots: Different approaches in evolutionary robotics. In R. Brooks and P. Maes, editors, *Artificial Life IV*, pages 190–197. MIT Press/Bradford Books, 1994.
15. A. Thompson. Evolving electronic robot controllers that exploit hardware resources. In F. Moran, A. Moreno, J.J. Merelo, and P. Chacon, editors, *Advances in Artificial Life: Proc. 3rd European Conference on Artificial Life*, pages 640–656. Springer-Verlag, Lecture Notes in Artificial Intelligence 929, 1995.

This article was processed using the LaTeX macro package with LLNCS style