Harnessing Morphogenesis

of Morphogenesis that developmental biologists are only beginning to uncover. In itself this is no bad thing, but since no entirely satisfactory method of encoding really complicated neural networks in a compact way has been discovered, this paper represents a step towards exploiting the encoding scheme used by mother nature.

The inspiration behind the encoding scheme described in this paper comes from biology, but it was designed with evolutionary robotics firmly in mind, and not as a defensible model of biological development. The overall picture to be gleaned from the model, described in the next section, will at least be reminiscent to the developmental biologist of what actually takes place during Morphogenesis, but where precise functional details of a particular developmental mechanism were unavailable, or where the biological details appeared to conflict with the goals of evolvability and flexibility of design necessary to any encoding scheme, alternative mechanisms were implemented. Biological terms are used extensively throughout the paper. Unless stated otherwise they should be seen as explanatory tools rather than strong references to their biological counterparts.

2 An Overview of the Developmental Model

This section gives a general overview of the encoding scheme in three stages. First it describes what takes place at the level of the genome, then how this controls the behaviour of individual cells, and finally how a multicellular 'organism' develops from a single cell.

In the current scheme, a genome consists of a single string of what might be thought of as base-pairs of nucleic acids (i.e. one of four characters). The start of each gene on the genome is identified by a certain pattern of preceding characters, similar to the TATA box on a real genome (see [6] for a good introduction to developmental biology). Genes are of fixed length. Each gene is responsible for the production of a particular protein and in turn is regulated by certain combinations of proteins within the cell. What is important to realise at this stage is that genes create proteins that regulate genes (see [15] for a beautifully clear example of this), forming genomic regulatory networks (GRNs) that control the entire behaviour of each cell during development. These are best thought of as independent dynamical systems within each cell, capable of being knocked from one basin of attraction into another by internal and external stimuli, following independent trajectories through state space [1] (for full details see Section 4.1).

The proteins within each cell are divided into different classes. Each has a unique effect on the gross behaviour of the cell. The ability of a given protein to perform its role within its class depends on its 'shape' (for a full explanation of each class and how they interact with each other see Section 4.3). Signal proteins diffuse out of one cell and into another, allowing cells to influence each other at a distance. The *direct* consequences of this influence may take one of two forms, either perturbing the cell's internal dynamics (turning certain genes 'on' and others 'off') or applying a force on the cell body towards or away from the protein source. In practice, there are many different signal proteins, from many different sources, entering each cell at any given time, and each cell will only respond to a subset (depending on the state

[1] For a fuller account of this interpretation and, to my knowledge, the first exposition of genomic functionality as a regulatory network, see [13].

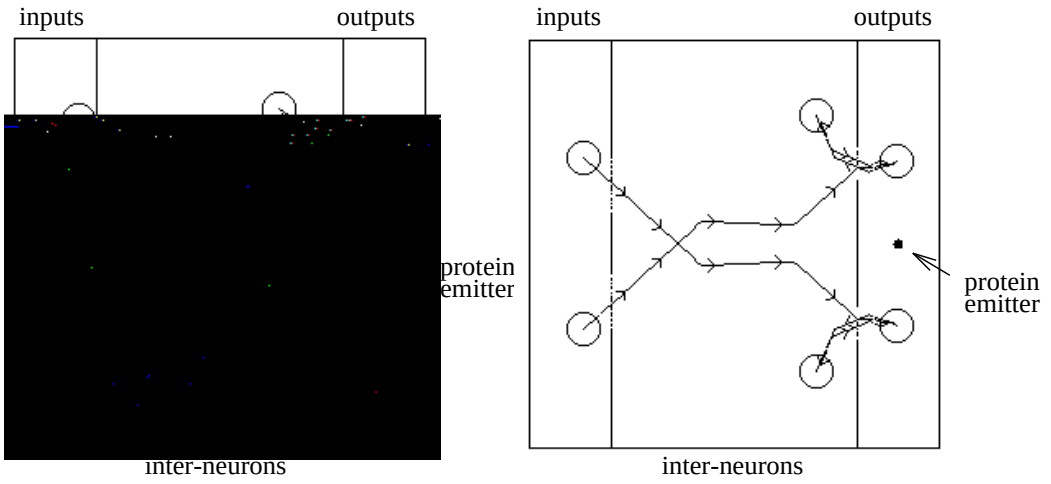


Figure 1: *Two evolved neural networks. The network on the left exhibits corridor following behaviour, while that on the right exhibits obstacle avoiding behaviour. The input region of the developmental environment was divided into eight sub-regions, from top to bottom, corresponding to the eight infra red light sensors on the Khepera robot. The output region was divided into two sub-regions, corresponding to the left and right motors of the Khepera.* small

of its internal dynamics). This means that different cells may behave differently in response to identical external stimuli, and also that different cells may behave identically in response to different external stimuli.

Initially a single cell is placed in an environment containing a number of strategically placed extra-cellular signal protein sources. These fixed sources provide a reference for the developmental process. After the initial cell begins to divide, individual cells divide and move, as part of one big dynamical system coupled together by signal protein interactions, until, eventually, they differentiate (see section 4.4). When this occurs, a number of dendrites grow out from each cell guided by 'growth cones' sensitive to unique combinations of signal proteins. On contact with another cell, a synaptic connection is formed (see Section 4.5). After every cell has differentiated and every dendrite has either connected or died, thresholds and weights are assigned to each cell and dendrite respectively, and inputs and outputs are assigned to cells which lie in specific regions of the developmental environment, to form a neural network ready for testing (see section 4.6).

3 Preliminary Experiments

Before going into the details of the encoding scheme, I will first outline the results of preliminary experimenlieuli,

to robot. Both experiments involved a population of one hundred character strings, each of which was 5000 (initially random) characters long. The genetic algorithm in both cases was based on a simple generational model with rank-based selection. Crossover and mutation were the only genetic operators used. Crossover happened at every 'breeding' and the mutation rate was set at 0.002 mutations per character on the genome (about 12 mutations per genome on average).

In the first experiment, the robot was placed at one end of a long corridor with many bends, and neural networks were evolved that could guide it down the corridor without crashing into the walls. The fitness function (similar to that used in [3]) returned the normalised product of three terms: one for going as fast as possible, one for going as straight as possible and one for staying away from walls. By generation 10 networks had evolved that could guide the robot down the corridor. The network displayed in Figure 1, which was the fittest individual of generation 50, could successfully guide the robot down the corridor without it touching the walls. The experiment was run for a total of 500 generations but no real improvement was made after the fiftieth generation.

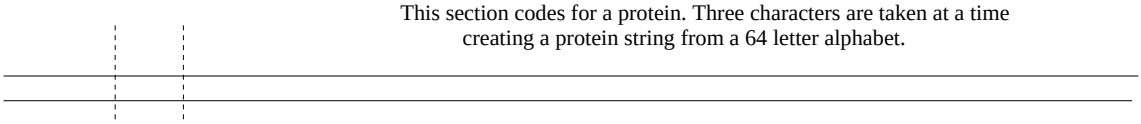
In the second experiment, the robot was placed in a rectangular environment containing many small cylinders. The fitness function was the same as that used above. Again, by generation 10, neural networks had evolved which evoked obstacle avoiding behaviour in the robot. The network displayed on the right hand side of Figure 1 is again the fittest member of the 50th generation. This network proved to be very good at obstacle avoidance. However, since it only makes use of two of the available eight sensors on the robot, the robot has several 'blind spots', including straight ahead for small objects.

4 The Encoding Scheme in Detail

The reader who is not interested in the details of the algorithm that converts character strings to artificial neural networks may skip this section if they want. They should, however, be warned that a full appreciation of the issues raised in Section 5 is not possible without a more involved understanding of the encoding scheme.

4.1 The Genomic Regulatory Network (GRN)

Proteins regulate genes which produce proteins. In other words proteins regulate other proteins. Each unit in the GRN corresponds to exactly one protein in the cell and the pattern and nature of links between units is defined by which protein(s) regulate which other protein(s). The activity of each unit equals the intra-cellular concentration of its particular protein (which is equal in turn to the activity of the gene that produces it: maximum 1.0, minimum 0.0). In reality, it may (and usually does) take the *pre ence* of several proteins and the *ab ence* of several others to activate a particular gene (thus increasing the cellular concentration of the protein it encodes for). There are usually several links fanning into each unit in the GRN, each with a weight between 2.0 and -2.0, and each unit has a unique threshold that sets the lower bound of the linear threshold activation function.


This section codes for a protein. Three characters are taken at a time
creating a protein string from a 64 letter alphabet.

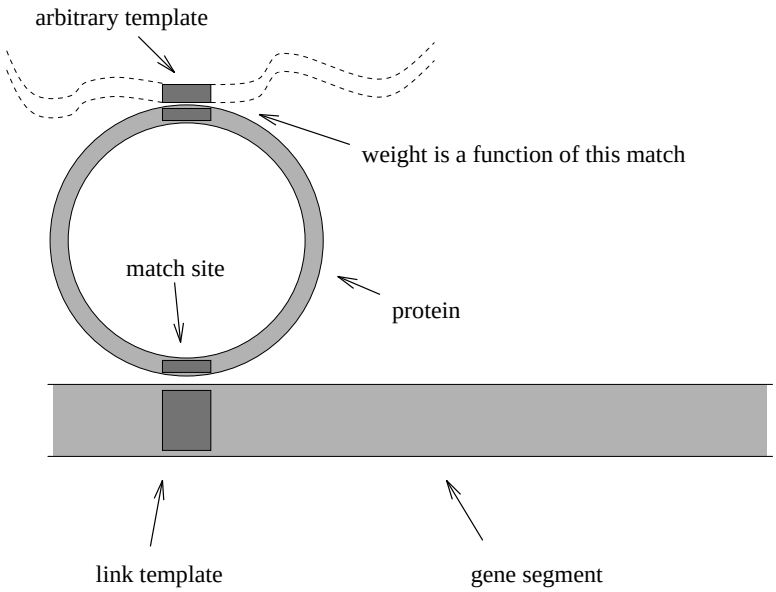


Figure 3: *A protein has been rotated until the match between the link template on the gene and the corresponding protein segment goes over a certain threshold. The contribution this protein makes to the regulation of the gene is then calculated from matching the diametrically opposite side of the protein with an arbitrary fixed template.*

and 64, and then to a decimal between plus and minus one.

In the nucleus of a real cell, which and how many genes any given protein regulates depends on that protein's shape. This shape is arrived at by a complicated folding process, the biological intricacies of which are poorly understood. In the model described in this paper, each protein is a circular string. To work out whether any given protein plays a part in

directly from the state of that cell's GRN. However, the activations of those genes (and hence the corresponding units in the GRN) within that cell that are regulated by signal proteins are a function of the amount of signal protein diffusing *into* that cell through the cell wall. See Section 4.4 for an explanation of how the amount of signal

a linear threshold function with lower bound set by the threshold on each node. All internal variables are then updated.

Forces on each cell are calculated by the application of mover proteins to the signal proteins in each cells locality. Each mover protein causes a force proportional to, and in the direction of the maximum gradient of, the weighted sum of the local chemical concentrations of the members of its particular subset of signal proteins. The weights in this sum correspond to the mover protein's affinities for each of the members of its subset. The cell moves in the direction of the resultant of these forces.

If any internal variable (for splitting or for differentiation) has gone over its threshold, then the appropriate action takes place. Note that when a cell splits, each daughter cell is identical to the mother cell in all but a slight positional displacement at right angles to the axis of cleavage.

When the internal differentiation variable within a cell goes over a threshold, the cell differentiates. At this point the cellular concentrations within the cell become fixed, including the levels of signal proteins output from that cell.

4.5 Differentiation and Dendrite Growth

At differentiation the number of dendrites that will grow from a cell is calculated. This is the number of dendritic proteins whose intra-cellular concentration lies above the threshold associated with that protein (see Section 4.3). As with mover proteins, each dendritic responds to its own particular subset of signal proteins. Growth starts from the side of the cell where the weighted sum of the concentrations of each signal protein in this subset is greatest.

Dendrites are guided by a trident shaped 'growth cone' with three sensors, one at the tip of each fork, each responsive to the dendritic protein's subset of signal proteins. At each time step the concentrations at each sensor are calculated, and the position of the base of the trident is updated to that of the trident 'tip' with the highest concentration. In this way the dendrite is steered, at a fixed speed, towards the local maximum of the weighted sum. This will usually be another cell, at which point the dendrite forms a connection with that cell and stops. However, since the sum is weighted, the dendrite may actively steer away from signal protein sources as well as towards them. This creates two sorts of problems. One, dendrites may grow off to infinity, and two, they may go into 'orbit' around local chemical minima. For this reason there is a maximum length to which a dendrite may grow before it is said to be dead.

4.6 Interpreting a Neural Network

Once all cells have differentiated, and all dendrites have either connected or died, the finished structure may be interpreted as an artificial neural network (ANN) which can then be used to control a robot. The architecture and connection matrix of the ANN is taken directly from that of the developed organism. Activation is interpreted to flow in the direction of dendrite growth. Each link is either inhibitory or excitatory (depending on the results of a template match. See Section 4.3). The weight on each link is calculated from the concentrations of the signal protein subset attributed to the dendritic protein responsible. The weighted sum of these

concentrations, at the point where the 'growth cone' connects, is divided by the maximum possible value of this weighted sum (i.e. as if all the concentrations were 1.0) to give a number between 0 and 1. This is then scaled appropriately.

Thresholds on the units in the ANN are calculated directly from the concentration levels of threshold proteins in the corresponding cells (see Section 4.3). Input and output units to the ANN correspond to those cells that end up in certain regions of the developmental environment. See Section 3 for an example of how this is done.

5 Issues of Evolvability

There are two properties *e ential* to an encoding scheme capable of evolving complex control architectures for robots. Firstly, it must be robust with respect to the genetic operators used by the genetic algorithm (cross-over, mutation, translocation, genome growth etc.), and secondly, it must be capable of 'subroutining', in the sense of encoding for repeated structure in a compact way. The encoding scheme described in this paper displays both of these properties, which are described below, and it is this that makes it potentially very powerful indeed.

5.1 Robustness to the Genetic Operators

If an encoding scheme is to work at all (i.e. to provide the basis for something more than random search) then it must be robust with respect to the genetic operators it is used with. It is crucial, if evolution is to progress, that 'fit' phenotypic traits are not destroyed by the breeding process. Ideally, with the exception of mutation, all genetic operators should cause as little phenotypic disruption as possible. The crossover operator is a special case, since it acts on two genotypes as opposed to one, but I would argue along with [7] that it is only meaningful, in an Evolutionary Robotics context, to use crossover in conjunction with converged populations.

The encoding scheme, reported here, is robust with respect to operators involving translocation and genome growth. This allows two things: firstly, genes that work in tandem (e.g. to create a 'fit' subnetwork in the phenotype) may be relocated next to each other on the genotype, thus minimising the chance of their separation by crossover, and secondly, open ended evolution becomes possible, where phenotypic complexity is not restricted by the original size of the genome. This robustness follows from the simple reason that, using template matching, parts of the genotype address other parts of the genotype (to form the links of the GRN) by the character sequences that occur there, and not, as in many encoding schemes, by a function of the region's location on the genome. If these sequences of characters are relocated

small scale changes to the GRN that controls development may nevertheless produce large scale changes in the phenotype (whole new portions of network may grow, or sub-structures may get repeated, see below). Also a small change to the GRN may do nothing but slightly alter one parameter of the phenotype. Slight mutations to the GRN, then, provide the right range of changes in the phenotype.

What is less obvious is why slight changes to the genotype, in the encoding scheme reported here, should result in slight changes to the GRN. Section 4.2 explains how the GRN is decoded from the genome using a variety of template matching operations. The problem is that if two or more templates overlap then a single mutation at this point may result in a disproportionately large change to the GRN. In order to minimize this possibility, the template matching routines were designed to extract the maximum amount of information from templates that were as short as

tal environment. However, cells elsewhere in the developmental environment may respond to different signal proteins subsets and may, therefore, develop asymmetrically. The final developed network will be asymmetric as a whole but with two identical subnetworks.

6 Conclusions

This paper represents an attempt to harness the developmental power of morphogenesis to the still young discipline of Evolutionary Robotics. A biologically inspired

[7] I. Harvey. Species adaptation genetic algorithms: the basis for a continuing saga. In F. J. Varela and P. Bourgine, editors, *Toward a Practice of Autonomous Systems: Proceedings of the First European Conference on Artificial Life*, pages 346–354, Cambridge, Massachusetts, 1992. M.I.T. Press / Bradford Books.

[8] I. Harvey. *The Artificial Evolution of Adaptive Behaviour*. PhD thesis, University of Sussex, 1993.

[9] I. Harvey, P. Husbands, and D. Cliff. Seeing the light: Artificial evolution, real vision. In *Proceedings of the Third International Conference on Simulation of Adaptive Behavior*, 1994.

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

[11] 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.

[12] K-Team. Khepera users manual. EPFL,Lausanne, June 1993.

[13] Stuart A. Kauffman. *The Origins of Order*. Oxford University Press, 1993.

[14] S. Nolfi, O. Miglino, and D. Parisi. Phenotypic plasticity in evolving neural networks. In *Proceedings of the PerAc'94 Conference*. IEEE Computer Society Press, 1994.

[15] M. Ptashne. *A Genetic Switch*. Cell Press, 1986.

[16] J. Vaario. From evolutionary computation to computational evolution. *Informatica*, 1994.