Developmental Neural Networks for Agents

Andy Balaam

CCNR, University of Sussex

Abstract. A system for generating neural networks to control simulated agents is described. The networks develop during the lifetime of the agents in a process guided by the genotype and affected by the agent's experience. Evolution was used to generate effective controllers of this kind for orientation and discrimination tasks as introduced by Beer. This scheme allows these behaviours to be generated quickly and effectively and may offer insights into the effects of developmental processes on cognition. For example, development may allow environmental regularities to be recognised without genetic prespecification. Possible future research into the abilities of these controllers to adapt to radical changes and to undertake widely varying tasks with a single genotype is described.

1 Introduction

Much of the agent-based modelling in Adaptive Systems research involves using evolution to find genetic 'blueprints' for controllers which perform a given task. A natural extension to this idea is to incorporate a developmental process that allows one to use evolution to find genetic 'recipes' for generating useful controllers.

The use of a developmental process to generate phenotype and behaviour from a genotype may have several advantages over the direct specification of the phenotype in the genotype. It may allow some of the work required to perform a task to be done during the development process (responding to the regularities found in a given environment, perhaps) rather than requiring evolution to solve these problems. It may also offer the ability to produce complex behaviours later in the lifetime of the agent, building upon simpler behaviours acquired earlier. In the longer term there are rich possibilities for this work to feed into biological and psychological studies of development and cognition.

Thus it is interesting to investigate developmental processes in the design of controllers. This paper presents the first steps in the design of a developmental controller which seeks to build upon the work already done in agent-based evolutionary modelling in two ways: first, it is based upon continuous-time recurrent neural networks [1] which have been studied extensively, and second it deals with 'minimally cognitive' tasks as defined by Beer [1, 18]. By taking this approach we are able to understand these new systems using our understanding of existing systems.

Since systematic study is essential in exploring a new area, so far these controllers have been evolved to perform extremely simple tasks. However, already

some very interesting consequences of the developmental process have been observed. In section 6 it will be argued that the abilities and scientific interest of these controllers may be expected to increase as the complexity of the tasks they are required to perform increases.

2 Background

2.1 The importance of development in living systems

Recent work in neuroscience has uncovered a much greater incidence of structural change (including significant growth and death of neurons) in brains than previously expected. This change occurs even after brain development is complete. It is increasingly believed that this kind of structural change is vital for cognition and memory.

Recent studies have shown that newly generated neurons not only appear in the brain, but that they become involved in its functional activity. For example, van Praag et al [19] characterised new neurons in adult mice over time after their appearance and found that they developed morphologies similar to those of mature neurons. They were able to show strong evidence that these new cells received synaptic input from their neighbours and were functionally incorporated into the hippocampal network.

Meanwhile, momentum for the general view that change is vital and pervasive in adult brains is continuing to grow [14]. For example, Ivanco and Greenough [11] argue that the changes brought about by learning and experience are exhibited as physical changes indicating functional reorganisation, and that the mechanisms that bring these changes about may be the same mechanisms that repair tissue after damage to the brain.

The use of neural network models by neuroscientists to investigate the role of structural change in real brains is becoming common ([4], [13],[15],[16], [17]).

If cell growth and death and synaptic plasticity are important to the cognitive capabilities of animals, it is reasonable to assume that in order to produce cognitive capabilities in artificial agents we will need to allow for analogous processes of structural change.

2.2 Adaptive systems approaches

In the early 1990s several researchers approached the problem of developmental neural networks. Most of the work done at this time used string and graph rewriting grammars to model the development process. Perhaps the most successful of these models was that of Gruau [7–9], who used a graph rewriting grammar coupled with a tree-like genotype to create modular neural networks that were grown in a development phase before the lifetime of an agent. Thus the development was decoupled from the experience and sensory capabilities of the agent and encapsulated in a very abstract mathematical model.

In 1995 Jakobi [12] designed a developmental neural network scheme involving models of DNA molecules, protein transcription and diffusion and a genomic

regulatory network. A controller was seeded at the beginning of an agent's lifetime with a single unit and others grew according to the rules of the genotype and protein interactions. This development occurred within a two-dimensional controller space, and took place before the agent's lifetime began.

After several years with very little activity, momentum has been growing recently behind research into developmental controllers.

Elliot and Shadbolt [3] have successfully shown that obstacle avoidance behaviour can be generated in a real robot by the use of a hand-designed (as opposed to evolved) developmental system that models some of the processes found in neuroscience.

There is a growing level of interest in plasticity in neural networks, especially in terms of their synaptic connections. Several pieces of work have been done using Floreano and Mondada's plastic neural networks [2, 5, 6], to investigate the potential benefits of this limited form of structural change in controllers.

3 Method

The environments and tasks modelled are designed to be identical to some of those used by Beer [1] in his research into minimally cognitive behaviours. In addition, the controllers used are closely linked with Beer's continuous-time recurrent neural networks (CTRNNs).

In parallel with the developmental experiments, replications of Beer's experiments (using CTRNN controllers) on the tasks used were undertaken. These may serve as controls to quantify the performance and the behaviour of the controllers.

3.1 Continuous-time Recurrent Neural Networks

Neurons in both the developmental networks and the standard CTRNNs are governed by the following equation: [1]

$$y_{i,t} = y_{i,t-\Delta} + \left(\frac{\Delta}{\tau_i}\right)\left(-y_{i,t-\Delta} + \sum_j w_{j,i} z_{j,t-\Delta} + S_i\right)$$

where $\sum_j$ denotes the sum over all neurons j connected to neuron i, and

$$z_{i,t} = \frac{1}{1+\exp(-(y_{i,t}+b_i))}$$

and $y_{i,t}$ is the cell potential of neuron i at time t, Δ is the time step size being used (0.1 in this work), τ_i is the time constant of neuron i, $z_{i,t}$ is the firing rate of neuron i at time t, $w_{j,i}$ is the weight of the connection from neuron j to neuron i, S_i is the amount of sensory input to neuron i and b_i is the bias of neuron i.

3.2 Developmental Neural Networks

The design of the developmental controllers is intended to strike a balance between biological inspiration and pragmatic assumptions about what kind of controllers will generate useful behaviours in a reasonable time.

The concepts inspired by biological development are: growth from smaller towards larger networks, an identical genotype for each unit and the use of simulated chemical gradients that specify which part of the genome is relevant to a given unit.

The general scheme is as follows: the controller is located within a two-dimensional space. Certain areas of the space correspond to the agent's sensors (neurons within those areas receive input from the corresponding sensor) and certain areas correspond to motors (the firing rates of neurons in these areas affect the action of the corresponding motor) as illustrated in figure 1. The controller is seeded at the beginning of the agent's life with one neuron per sensor on the agent.

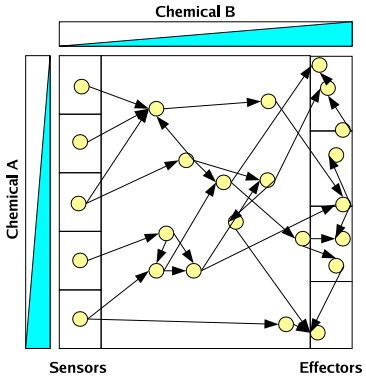

Fig. 1. The controller is a two-dimensional space with two chemical gradients over it, and specific sensor and effector regions.

There are two chemical gradients across the space, running horizontally and vertically. When a neuron is created, the levels of chemicals locally and the parent neuron's cell potential (if it has a parent) serve as inputs to functions the output of which provide the new neuron's time constant, bias, and energy. The shape of those functions is defined by the genotype.

At every time step the neuron's cell potential and the local chemical levels are fed into functions defined by the genotype to give the amount by which that neuron's 'growth sum' will change. When the growth sum goes beyond a threshold - if the neuron has any remaining energy - the neuron grows a new 'child' neuron, using functions defined by the genotype to determine the angle and distance at which to grow the new neuron. A connection is created linking from the parent to the child, and its weight is found using a function defined by the genotype. The energy of the parent neuron is decreased by 1 whenever it grows a child. If a neuron already exists at the point in the space where a new one is to be grown, a link is formed to the existing neuron instead.

The functions mentioned above are designed to approximate an arbitrary function using the following form:

$$p_n = \sum_{i=1}^{N_c} \sum_{j=1}^{N_s} (a_{i,j,1} \sin (a_{i,j,2} x_i + a_{i,j,3}))$$

where p_n is a property of the neuron ($n = 1, \ldots, 7$), $a_{i,j,k}$ is a genetically-determined value, x_i is the local concentration of chemical i or the cell potential, N_c is the number of chemicals + 1 for cell potential (= 3 in this work), N_s is a constant (10 in this work).

The genotype is a list of 630 real values, since $7 \times N_c \times N_s \times 3 = 630$. The 7 occurs because there are 7 properties: bias, time constant, energy, growth increment, direction of growth, distance to grow and new connection weight. So the genotype may be thought of as a vector function, taking 3 inputs (chemical 1, chemical 2 and cell potential) and having 7 outputs.

3.3 Minimally Cognitive Behaviours

The environment and agent are designed to be exact replicas of those used in [1] including relative distances and speeds.

The environments in which the agents behave are extremely simplified situations designed to encapsulate certain fundamental elements of cognition. Agents themselves are circular and able to move only left and right at the bottom of a two-dimensional environment. Objects fall from above the agents and tasks require either 'catching' (matching horizontal positions when vertical positions are matched) or 'avoiding' objects in different circumstances.

Agents have distance-sensitive ray sensors (analogous to infra-red sensors on real robots) which produce large input when an object crosses a ray close to the agent and small input when it crosses far away. These sensors are arranged in a fan shape pointing out from the top of the agent over an angular range of $\pi/6$ (figure 2).

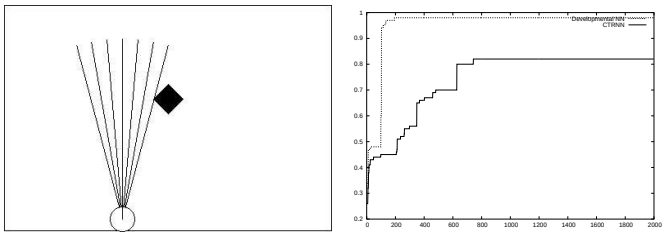

Fig. 2. Left: The agent has ray distance sensors and may move left and right in the environment. Right: Fitness vs. generation. The best fitness achieved with a developmental controller in the orientation experiment far out-performed the best achieved by a CTRNN.

3.4 Genetic Algorithm

A generational, asexual genetic algorithm using rank selection with elitism (4% elitist fraction) on a population size of 50 is used, with real-valued genotypes. In CTRNNs mutation is performed by adding a random displacement vector whose direction is uniformly distributed on the M-dimensional hypersphere and whose magnitude is a Gaussian random variable with mean 0 and variance 10. In developmental networks mutation involves randomly selecting 5 of the 630 real values on the genotype and randomising them within their allowed ranges.

The fitness of an agent over several presentations is calculated as follows:

$$F = 0.45f_m + 0.45f_w + 0.1f_c$$

where f_m is the mean fitness over the presentations, f_w is the fitness achieved in the worst (lowest fitness) presentations, and f_c is the fitness awarded for attributes of the controller. f_m, f_w and f_c are $\in [0:1]$ and thus $F \in [0:1]$.

In CTRNN controllers f_c is always equal to 1, and in developmental controllers f_c is equal to the mean (over the presentations) of the quantity x/W where x is the horizontal position of the rightmost neuron in the controller at the end of the agent's lifetime and W is the width of the controller space. So agents are rewarded for growing from the seed neurons on the left towards the motor region on the right.

4 Experiments and Results

The simulated environment was 400 wide by 275 high (the units are arbitrary) and contained the agent (which was circular with radius 15), and objects: circles of radius 13 and squares (called 'diamonds' since they were rotated by 45°) of side 26.

Orientation In the first experiments circles were dropped at varying speeds and directions from the top of the environment and agents were required simply to orient themselves to the circles, catching them. Fitness for a single presentation was awarded as $F_p = 1 - \frac{d}{W}$ where d was the distance between the agent and the circle at the end of the presentation and W was the width of the environment (i.e. the maximum possible distance). The final fitness of an agent was found by combining fitnesses for each presentation as specified in section 3.4.

The developmental controllers performed at least as well at this task as standard CTRNN controllers. In fact, over the course of 20 evolutionary runs of each type (2000 generations in each), the maximum fitness acquired for developmental controllers (0.98 - the maximum possible was 1) was significantly higher than that acquired for CTRNNs (0.82), and 8 of the 20 developmental runs achieved a fitness higher than 0.82.

Discrimination The second set of experiments involved the agents discriminating between objects of different shapes. Circular or diamond-shaped objects were dropped vertically from the top of the environment at different horizontal positions and fitness was awarded as follows:

$$F_p = \begin{cases} 1 - \frac{d}{D} & \text{when the object was a circle} \\ \frac{d}{D} & \text{when the object was a diamond} \end{cases}$$

where d was the distance between the agent and the object at the end of the presentation (capped to be $\leq$ D) and D was a maximum distance less than the width of the environment.

In these experiments the developmental controllers performed reasonably well, with the best agents, when examined manually, clearly correctly discriminating in 15 of 20 presentations. However, CTRNN controllers perform better at this task, with the best controllers correctly discriminating in 18 of 20 presentations.

5 Analysis

A number of interesting interactions and dynamics arise in the evolved solutions to the tasks described above. These fall into several categories:

Specialisation A potential advantage of the use of developmental controllers over standard CTRNNs is that development allows the agent to specialise according to the environment in which it finds itself. Of course, in the tasks so far studied the need for specialisation does not appear pressing to an observer, but specialisation did occur in some cases. For example, as shown in figure 3 the same agent (in this case in the discrimination task) developed different controller structures depending on the environmental conditions. The behaviours exhibited by this agent are shown in figure 4.

While the tendency to specialise may have negatively impacted the performance of agents in this simple task (due to problems with consistency), it seems likely that this feature will be extremely useful when tackling more difficult and complex tasks (see section 6).

Indirect genotype-phenotype mapping The mapping between genotype and phenotype in the simulations described is indirect, in contrast with much of the work being done in agent-based robotics today. The advantages of this, beyond its biological inspiration, are that it allows smooth changes of interesting kinds (such as rotations of whole sub-trees of neurons due to the change in angle of a single growth), it allows for modularity and genotype reuse (for example similar structures may be developed at the top and bottom of the controller if the vertical chemical gradient is insignificant in a certain area), and most of all it is scalable to large numbers of neurons without a corresponding increase in genotype length. This allows the evolutionary process to find solutions with

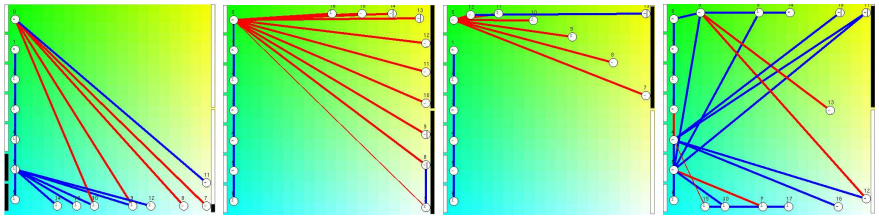

Fig. 3. The controller structures grown by a single agent under different environmental conditions. The two structures on the left show the controller when circles are dropped at different horizontal positions. The two on the right show what happens when diamonds are dropped. Circles must be caught and diamonds avoided. Sensor neurons (with which the controller is seeded at the beginning of a lifetime) appear on the left, and effector neurons on the right.

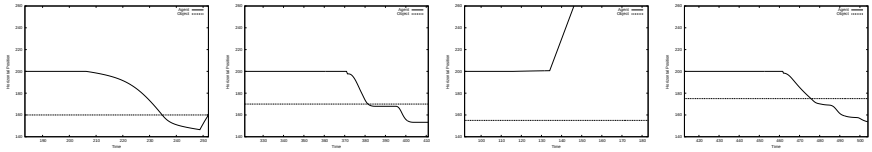

Fig. 4. The behaviour of the agent with the controller illustrated in figure 3. The horizontal line shows the horizontal position of the falling object over time. The other line shows the position of the agent.

appropriate numbers of neurons without this constraint being imposed from outside.

Neutrality and rich fitness landscapes The fitness landscape of these controllers contains a great deal of neutrality; for example, the growth of a neuron into an area of the space which does not affect the motors has no effect on behaviour.

Moreover, the fact that solutions were found to the problems studied in reasonable timescales implies that the landscape is quite rich with effective solutions, despite the relatively vast size of the genotype (630 values as compared with 12 and 47 values in the CTRNN versions of the orientation and discrimination tasks respectively). This suggests that the space of all developmental controllers is quite densely packed with interesting behaviours.

6 Conclusions and Future Work

The experiments described here represent the first steps in the design of developmental controllers that satisfy two key criteria: that they should encapsulate some of the properties of development that are useful for producing complex and

interesting behaviours, and that they should be simple enough to allow for evolution of such interesting behaviours in a reasonable time. This early work goes towards justifying the assertion that the latter criterion may be satisfied, since the controllers evolved to a higher fitness than CTRNNs under identical conditions, and often achieved this high fitness in fewer generations. In regard to the former criterion, some of the potential of development has been shown, notably the development of controllers specifically suited to particular environmental conditions, but future work will be aimed at showing further that development is useful in producing more interesting behaviours.

Initial directions will be towards understanding the capabilities of these controllers as they are by testing them with more complex environments and tasks, and then working towards increasing the power of the controllers by discovering how to maximise the advantages of developmental processes and minimise the disadvantages. Adjustments are likely to be required, especially to the nature of the functions described in section 3.2. Some of the drawbacks of the current design are becoming clear, including a high chance of catastrophic mutations, and local fitness maxima caused by the unidirectional nature of the development process.

Development in biological organisms can allow the regularities in the environment to be automatically incorporated during the organism's lifetime into its cognitive faculties, rather than predicted and prespecified in the genotype. This effect may allow artificial developmental controllers to lift some of the cognitive burden away from direct evolutionary control, into the developmental process. Thus environments with existing but varying regularities could be used to test whether developmental controllers can more easily take advantage of these regularities than conventional controllers. An extreme case of this would be asking a controller to perform two different behaviours in two completely different environments, distinguishable only by the sensory input the agent receives in each environment.

Another way in which developmental processes are of benefit is in adaptation to radical changes to an agent or its environment. For example as an organism grows larger over its lifetime developmental processes allow it to cope with this change. Interesting experiments may be imagined involving predictable but radical changes to an agent's environment. For example, an agent that first has to navigate a corridor before emerging into an open space where it has to perform some other task (approaching a certain area, say) may well benefit from being fitted with a developmental controller that can change in structure at the crucial moment to perform a different behaviour. Developmental processes may also benefit agents that need to adapt to unpredictable changes as well.

Along with the goal of producing controllers capable of complex and interesting behaviours, a long term goal of this work is to shed light on the relationship between development and cognition. If the results of experiments such as Held's [10] demonstration of links between behaving and learning in kittens could be reproduced in an artificial system, the benefits of artificial systems could be put to good use in understanding the nature of such systems in general terms.

References

1. Beer, R.D. (1996) 'Toward the evolution of dynamical neural networks for minimally cognitive behavior' In P. Maes, M. Mataric, J. Meyer, J. Pollack and S. Wilson (Eds.), *From animals to animats 4: Proceedings of the Fourth International Conference on Simulation of Adaptive Behavior* pp421-429. MIT Press.
2. Di Paolo, E.A. (2000) 'Homeostatic adaptation to inversion of the visual field and other sensorimotor disruptions' *Proc. of SAB 2000.*
3. Elliot, T. and Shadbolt, N. (2001) 'Growth and repair: Instantiating a biologically inspired model of neuronal development on the Khepera robot' *Robotics and Autonomous Systems* 36 pp149-169.
4. Elman, J. (1993) 'Learning and development in neural networks: The importance of starting small' *Cognition* 48 pp71-99.
5. Floreano, D. and Mondada, F. (1996) 'Evolution of plastic neurocontrollers for situated agents' In Maes, P., Mataric, M., Meyer, J., Pollack, J., Roitblat, H., and Wilson, S., editors, *From Animals to Animats IV: Proceedings of the Fourth International Conference on Simulation of Adaptive Behavior* MIT Press-Bradford Books, Cambridge, MA.
6. Floreano, D. and Mondada, F. (1998) 'Evolutionary neurocontrollers for autonomous mobile robots'. *Neural Networks* 11, pp1461-1478.
7. Gruau, F. and Whitley, D. (1993) 'Adding learning to the cellular development of neural networks: Evolution and the Baldwin effect' *Evolutionary Computation* 1, 3:213234.
8. Gruau, F. (1994) 'Neural Network Synthesis Using Cellular Encoding and the Genetic Algorithm' PhD. Thesis, Ecole Normale Superieure de Lyon, Laboratoire de l'Informatique du Parallelisme (LIPIMAG).
9. Gruau, F.(1995) 'Automatic Definition of Modular Neural Networks' *Adaptive Behaviour* 3.2 pp151-183.
10. Held, R. (1965) 'Plasticity in sensory-motor systems' *Scientific American*, 213(5), 84-94.
11. Ivanco, T., and Greenough, W. (2000) 'Physiological consequences of morphologically detectable synaptic plasticity: potential uses for examining recovery following damage.' *Neuropharmacology* 39, pp765-776.
12. Jakobi, N. (1995) 'Harnessing Morphogenesis' Technical Report School of Cognitive and Computing Sciences, University of Sussex.
13. Karmiloff-Smith, A. (1992) *Beyond Modularity: A Developmental Perspective on Cognitive Science*, MIT Press.
14. Kolb, B., Forgie, M., Gibb, R., Gorny, G. and Rowntree, S. (1998) 'Age, Experience and the Changing Brain' *Neuroscience and Biobehavioural Reviews* 22.2 pp143-159.
15. Quartz, S. and Sejnowski, T. (1997) The neural basis of cognitive development: a constructivist manifesto' *Behavioural and Brain Sciences* 20 pp537-596.
16. Quartz, S. (1999) 'The constructivist brain' *Trends in Cognitive Sciences* 3.2 pp48-57.
17. Quinlan, P. (1998) 'Structural change and development in real and artificial neural networks' *Neural Networks* 11 pp577-599.
18. Slocum, A., Downey, D., Beer, R. (2000) 'Further Experiments in the Evolution of Minimally Cognitive Behaviour: From Perceiving Affordances to Selective Attention' in *From Animals to Animats 6: Proceedings of the Sixth International Conference on Simulation of Adaptive Behaviour.*
19. van Praag, H. et al (2002) 'Functional neurogenesis in the adult hippocampus' *Nature* 415, pp1030-1034.