Exploring developmental dynamics in evolved neural network controllers

Andy Balaam (a.j.balaam@sussex.ac.uk), CCNR, University of Sussex

Submitted for the degree of D. Phil.

University of Sussex

April, 2006

Declaration

I hereby declare that this thesis has not been submitted, either in the same or different form, to this or any other university for a degree.

Signature:

Abstract

Two new developmental neural network controllers for evolutionary robotics are described. These controllers present opportunities for experiments into many areas of development: the results of several experiments are described.

Relevant former work in evolutionary robotics and the natural sciences is reviewed. The motivations of the work are presented: to generate working examples of systems which may be said to exhibit development for use as tools in its study, and to explore various open questions involving development using these tools. Several open questions are presented, most of which involve suggestions that controllers which are constrained to be developmental may offer advantages (in terms of how easily they may be evolved to exhibit certain behaviours) over more traditional controllers.

The experimental work is divided into two parts. The first part describes a controller which has neurons located in a two-dimensional space growing during the robot's lifetime in processes affected by the experience of the robot as well as its genotype.

The second piece of experimental work describes a controller designed to put into practice the lessons learned in the first part: one which has a fixed size but allows for the growth and death of synapses as well as plastic changes in their weights.

Little evidence was found to support the ideas behind many of the open questions explored: in particular, the controllers with added developmental dynamics were not found to have any advantage over standard controllers in tasks involving flexibility to predictable changes or adaptability to different environments.

Discussion is offered of the reasons for the results found, the utility of the tools developed and lessons learned about evolving developmental controllers. Possible avenues for future work are suggested.

Acknowledgements

This work was funded by the Engineering and Physical Sciences Research Council, and I am extremely grateful to every tax-payer: may it always be that the bankers, estate agents and programmers pay for the expansion of human knowledge, and may I feel the same way when I am one of them.

I would like to thank my supervisors Ezequiel Di Paolo and Phil Husbands for shepherding me through triumphs and disappointments with enthusiasm where mine was lacking, reassurance when all seemed lost, ideas when I had none, and wisdom beyond anything I can hope to attain.

My friends in the CCNR, especially Ian Macinnes and Miguel Garvie, have made my time there very enjoyable - without our discussions about topics as varied as the existence of consciousness, the fatal flaws in our research, and the decline of society as exhibited by the invention of the electric toothbrush, I would not have made it. Many members of the CCNR have helped me out in many ways. I am particularly grateful to Paul Graham for his advice on statistics.

The most valuable things in my life are my family and friends: for all you give me and all you put up with from me, thank you.

If there were some way to repay my wife Pia for all that she has endured from me and given to me, I would do it, but all I can offer is the rest of my life, and regret that it will never be enough.

> So neither he who plants nor he who waters is anything, but only God, who makes things grow. *1 Corinthians 3:7*

> What does a man get for all the toil and anxious striving with which he labours under the sun? All his days his work is pain and grief; even at night his mind does not rest. This too is meaningless. *Ecclesiastes 2:22-23*

Contents

1 Introduction **9**

1.1 Overview . . . 9

1.2 What is development? . . . 11

1.2.1 Structural change: a two-tier system . . . 11

1.2.2 Development is not evolution . . . 12

1.2.3 Note on the word 'plastic' . . . 12

1.3 Summary . . . 13

1.3.1 Development of tools . . . 13

1.3.2 Exploring open questions . . . 14

1.4 Contributions of the thesis . . . 16

1.4.1 Tools development . . . 16

1.4.2 Investigating open questions . . . 16

2 Background in evolutionary robotics **17**

2.1 The overall research programme . . . 17

2.2 Previous work . . . 17

2.2.1 Experience-independent change . . . 18

2.2.2 Experience-dependent change . . . 19

3 Background in biology **23**

3.1 Development's role in learning and cognition . . . 23

3.2 Development and evolution . . . 24

3.2.1 Canalisation and genetic assimilation . . . 24

3.2.2 Acquired and inherited characteristics . . . 25

3.3 A live debate: constructivism and selectionism . . . 26

4 Motivation and Open Questions **28**

4.1 Motivation - why study development? . . . 28

4.1.1 Are CTRNNs developmental enough? . . . 29

4.1.2 To learn about how it works . . . 29

4.1.3 To build better robots . . . 30

4.2 Open Questions . . . 32

5 Methods **35**

5.1 Evolutionary robotics . . . 35

5.1.1 Agent-based simulations . . . 35

5.1.2 Artificial evolution and genetic algorithms . . . 36

5.1.3 Fitness functions . . . 37
5.1.4 Fitness landscapes and evolvability . . . 37
5.1.5 Studying evolved individuals . . . 39
5.2 Continuous-time recurrent neural networks . . . 39
5.3 Minimally cognitive behaviour . . . 41
5.4 Statistics . . . 42

6 Part 1 - Chemical-guided growth networks 44
6.1 Motivation . . . 44
6.2 Methods . . . 45
6.2.1 Robot and environment . . . 46
6.2.2 Controller . . . 47
6.2.3 Genetic algorithm . . . 51
6.2.4 Orientation task . . . 51
6.2.5 Discrimination task . . . 52
6.2.6 Multiple discrimination task . . . 53

7 Part 1 - Results, Analysis and Discussion 55
7.1 Results . . . 55
7.1.1 Orientation task . . . 55
7.1.2 Discrimination task . . . 55
7.1.3 Multiple discrimination task . . . 58
7.2 Analysis . . . 63
7.2.1 Behaviour . . . 64
7.2.2 Neural mechanisms . . . 67
7.2.3 Morphogenetic mechanisms . . . 69
7.2.4 Conclusions . . . 72
7.3 Discussion . . . 72
7.3.1 Observations . . . 72
7.3.2 A two-tier process . . . 76
7.3.3 Navigating a rugged landscape . . . 76
7.3.4 Generating structure from a blank slate . . . 77

8 Part 2 - Synaptic growth and pruning networks 79
8.1 Motivation . . . 79
8.1.1 From part one to part two . . . 80
8.1.2 Comparison between controller types . . . 81
8.2 Methods . . . 82
8.2.1 Robot and environment . . . 82
8.2.2 Controller . . . 85
8.2.3 Genetic algorithm . . . 88
8.2.4 Phototaxis task . . . 88
8.2.5 Corridor following task . . . 90

8.2.6 Predictable change task . . . 92
8.2.7 T maze task . . . 94
8.2.8 Double T maze task . . . 95
8.2.9 Learning task . . . 97
8.2.10 Re-learning task . . . 97
8.2.11 Flexibility task . . . 98
8.2.12 Robustness to disruptions tests . . . 99
8.2.13 Robustness in larger fixed networks . . . 100
8.2.14 Synapse growth verses death comparison . . . 101

9 Part 2 - Results, Analysis and Discussion 102
9.1 Results . . . 102
9.1.1 Phototaxis task . . . 102
9.1.2 Corridor following task . . . 102
9.1.3 Predictable change task . . . 105
9.1.4 T maze task . . . 107
9.1.5 Double T maze task . . . 111
9.1.6 Learning task . . . 111
9.1.7 Re-learning task . . . 114
9.1.8 Flexibility task . . . 119
9.1.9 Robustness to disruptions tests . . . 119
9.1.10 Synapse growth verses death comparison . . . 121
9.2 Analysis . . . 125
9.2.1 Behaviour . . . 125
9.2.2 Controller . . . 126
9.2.3 Development . . . 132
9.3 Discussion . . . 133

10 Discussion 134
10.1 Development of tools . . . 134
10.1.1 Developmental Controllers . . . 134
10.1.2 How to evolve developmental systems . . . 136
10.2 Exploring open questions . . . 140
10.3 Further discussion . . . 144
10.3.1 Where is development useful? . . . 144
10.4 Growth and Pruning 90 . . . 146
10.5 Conclusions and future directions . . . 147

List of Figures

2.1 Genetic operations in Gruau's (1994) growing neural networks. 18
2.2 The protein matching stage of the model described in Jakobi (1995). 20
2.3 The last stages of development in the controllers used by Cangelosi et al. (1994). 21
2.4 The paths taken by two evolved individuals from Nolfi and Parisi's work. 22

5.1 A fitness landscape in a very simple example situation. 38
5.2 An example of the behaviour of a neuron in a CTRNN. 40
5.3 An agent performing a minimally cognitive behaviour. 42

6.1 The minimal simulation robot in its environment. 46
6.2 The chemical-guided growth network. 48
6.3 The genotype of the chemical-guided growth network. 49
6.4 The orientation experiment. 51
6.5 The discrimination experiment. 53

7.1 Fitness vs. generation for the orientation experiment. 56
7.2 Mean fitnesses in the orientation task. 56
7.3 Fitnesses in the orientation task. 57
7.4 Histogram of fitnesses in the orientation task. 58
7.5 Mean fitnesses in the discrimination task. 59
7.6 Fitnesses in the discrimination task. 60
7.7 Histogram of fitnesses in the discrimination task. 60
7.8 A CTRNN robot evolved for the discrimination task, performing the multiple discrimination task. 61
7.9 Fitness over the first 50 generations of the multiple discrimination task. 63
7.10 Fitness vs. generation for the multiple discrimination experiment. 64
7.11 The chemical-guided growth network evolved to perform multiple discrimination. 65
7.12 Behaviour of the chemical-guided growth network when either a circle or a diamond is present. 66
7.13 Mean fitness of the robot when the falling object is 'switched' part-way through its descent. 67
7.14 The controller of the robot at different times in its life. 68
7.15 The potential direction in which a neuron will grow if a neuron grows at a particular time. 69
7.16 The growth sum of a neuron over the robot's lifetime. 70
7.17 The cell potential of a neuron over the robot's lifetime. 71
7.18 The amount by which a neuron's growth sum is increased at a time step, against the cell potential of the neuron. 71

7.19 The controller structures grown by a single robot under different environmental conditions. . . . 74
7.20 The behaviour of the robot. . . . 75

8.1 A single directional goal sensor. . . . 83
8.2 The robot's ray distance sensors. . . . 84
8.3 Forces on the body of the agent. . . . 85
8.4 Phototaxis task. . . . 89
8.5 Corridor following task. . . . 90
8.6 Predictable change task. . . . 93
8.7 T maze task. . . . 94
8.8 Double T maze task. . . . 96
8.9 Flexibility task. . . . 98

9.1 Mean fitnesses for the phototaxis task. . . . 103
9.2 Fitnesses in the phototaxis task. . . . 104
9.3 Histogram of fitnesses for the phototaxis task. . . . 104
9.4 Mean fitnesses for the corridor following task. . . . 105
9.5 Fitnesses in the corridor following task. . . . 106
9.6 Histogram of fitnesses for the corridor following task. . . . 106
9.7 Mean fitnesses for the predictable change task. . . . 107
9.8 Fitnesses in the predictable change task. . . . 108
9.9 Histogram of fitnesses for the predictable change task. . . . 109
9.10 Mean fitnesses for the T maze task. . . . 110
9.11 Fitnesses in the T maze task. . . . 110
9.12 Histogram of fitnesses for the T maze task. . . . 111
9.13 Mean fitnesses for the double T maze task. . . . 112
9.14 Fitnesses in the double T maze task. . . . 113
9.15 Histogram of fitnesses for the double T maze task. . . . 113
9.16 Mean fitnesses for the learning task. . . . 114
9.17 Fitnesses in the learning task. . . . 115
9.18 Histogram of fitnesses for the learning task. . . . 116
9.19 Mean fitnesses for the re-learning task. . . . 116
9.20 Fitnesses in the re-learning task. . . . 118
9.21 Histogram of fitnesses for the re-learning task. . . . 118
9.22 Mean fitnesses for the flexibility task. . . . 119
9.23 Fitnesses in the flexibility task. . . . 120
9.24 Histogram of fitnesses for the flexibility task. . . . 121
9.25 Robustness in the phototaxis task. . . . 122
9.26 Robustness in the T maze task. . . . 122
9.27 Fitness in the phototaxis task for different growth and death probabilities. . . . 123
9.28 Fitness in the T maze task for different growth and death probabilities. . . . 124
9.29 The growth and pruning controller. . . . 127

9.30 Standard deviation of final weight values in the growth and pruning controller. . . 128
9.31 The difference in synaptic strengths during two lifetimes of the growth and pruning controller. 129
9.32 The controller used for the analysis of the re-learning task. 130
9.33 Cell potentials of neurons throughout a lifetime. 131

List of Tables

7.1 Fitnesses in the orientation task. 57
7.2 Fitnesses in the discrimination task. 58

9.1 Fitnesses in the phototaxis task. 103
9.2 Fitnesses in the corridor following task. 106
9.3 Fitnesses in the predictable change task. 108
9.4 Fitnesses in the T maze task. 109
9.5 Fitnesses in the double T maze task. 112
9.6 Fitnesses in the learning task. 115
9.7 Fitnesses in the re-learning task. 117
9.8 Fitnesses in the flexibility task. 120
9.9 The performance of the re-learning robot when one or two neurons are removed. 127

Chapter 1

Introduction

1.1 Overview

How does the fact that natural behaving creatures grow from a single cell affect their behaviour? How does the process of development interact with evolution? Developmental systems continually cause wonder in human beings: the experience of parenting a child is often cited as the most profound experience a person can have. The secrets of development often seem more mysterious even than those of cognition - perhaps they contain the secrets of life itself: how it came about and how it so successfully filled the planet Earth.

This thesis describes an attempt to make a step very close to the beginning of the journey towards understanding development - a journey which may eventually lead to a greater knowledge of this process, and which may allow the use of this knowledge to build complex, adaptable and robust artificial creatures.

The thesis begins by examining the meaning of the word development for the purposes of this work before providing a summary of the research and its contributions, including the tools developed and open questions explored. It moves on to examine the relevant work which has been undertaken in the field of evolutionary robotics (chapter 2), and survey some relevant work in the biological disciplines (chapter 3). These chapters provide the context within which the work was completed, part of an ongoing research programme that is relatively young and must continue, mature and move forward a great deal before its ambitious goals are achieved. They describe in detail the motivation for the study of development, from very practical reasons about building better robots immediately, to the longer term aim of providing understanding which may feed into the biological sciences. The work of researchers on evolving robots whose controllers develop in various senses is described, and the degree of success they achieved in producing robots capable of performing a variety of tasks. In the biological realm, recent changes in perspective in neuroscience are described which suggest that the distinction between development and other processes such as cognition and learning may be largely artificial. Some philosophical issues over the unnatural concept of a separation between inherited and acquired characteristics, and the expected form that an evolved developmental process will take, are examined, looking at concepts such as canalisation and genetic assimilation. Finally, attention is turned to a debate in neuroscience about

whether brain development may be viewed as a Darwinian-like process.

The thesis continues by collecting together its motivations, stating the open questions which have been explored (chapter 4) and going over the generally-applicable techniques and concepts used in this work and in much of the similar work in this area (chapter 5). It reviews agent-based simulation, genetic algorithms, fitness functions, fitness landscapes, evolvability and studying artifacts produced by evolution before describing a commonly-used type of robot controller, the continuous-time recurrent neural network (CTRNN). After this it explains the methodology of studying minimally-cognitive behaviour, which is used in the following chapters.

The experimental work begins with part one, which is broken into two chapters (6 and 7) and describes the motivation, design, implementation, evolution and study of a new type of robot controller, based on an expansion of the CTRNN model, the chemical-guided growth network. The first part looks at the reasons for the decisions made in the design of the controllers and the tasks used to test them (based on the need for simplicity to allow successful evolution and the need to build on previous work), before describing these controllers in detail, with their neurons located in a two-dimensional space and growing neurons and synapses during the robot's lifetime in processes affected by the experience of the robot as well as its genotype. The results of the experiments are presented: it was found that these controllers may be successfully evolved to perform a very simple behaviour arguably of cognitive interest, but that the process was much harder than expected, since the developmental system was found to be more difficult to evolve than anticipated. Through a detailed study of the behaviour of one of these agents, and a general overview of the phenomena observed, some of the reasons for this difficulty are identified, most notably the difficulty of building up structure from a sparse network into a larger, more connected one, and the use of tasks which are unlikely to provide benefit for controllers capable of structural change.

Part two, again broken into two chapters (8 and 9) describes the way the lessons learned in part one were applied to the design of a new type of developmental controller, the growth and pruning network, which has a fixed size but allows for the growth and death of synapses as well as plastic changes in their synaptic weights. These controllers are described in detail, as well as the set of tasks that were designed to investigate some of the expected practical advantages of development, and the different roles of growth and death in this process. The results of these experiments are then presented, providing little support for many of the suggested answers to open questions, but notably showing that constraining some controllers to be developmental can improve robustness to certain disruptions. Again, study of an evolved individual provides insight into how developmental processes may work in practice under evolution, and some suggestions for ways to improve such processes are presented.

The thesis concludes in chapter 10 by gathering together the motivations, methodology, open questions, results and analysis undertaken and discussing wide-ranging topics such as the extent to which the ideas behind each open question are supported by the results found, the utility of the tools developed, practical ways to improve the chance of successfully evolving developmental robots, theoretical concepts behind the difficulties encountered and some of their possible solutions and suggestions that may be made in the debate about Darwinian processes in brain development. It concludes by looking at ways in which further research may move forward, and summarising

the advice that might be offered to those who wish to perform this work.

Before all of this may be attempted, it is important to examine what is meant by the word development. The following section answers this question for the purposes of this thesis.

1.2 What is development?

Section 1.2.1 explains that the definition of development here is structural change instantiating a two-tier system. A specific point about the clear distinction between development and evolution is then made in section 1.2.2.

1.2.1 Structural change: a two-tier system

For the purposes of this work, a developmental system is one which may be observed to consist of two tiers, a behavioural tier which instantiates the behaviour of an agent, and a structural change tier, which modifies the properties of the behavioural tier, changing its structure. The phrase structural change here is used as a short-hand for changes which modify the character of the dynamics of the behavioural tier, for example by altering the number of dimensions.

Since the division between tiers is subjective, this definition is dependent on the observer. Informally, the controller may be seen to consist of two controllers: a 'behaving controller' and a 'developing controller,' which adjusts the behaving controller. When these terms are used in this thesis they are intended to be understood as described here.

In an artificial neural network, the electrical and chemical interactions in neurons and synapses may be viewed as the behaving controller while changes to the properties of neurons and synapses, and their addition and removal may be viewed as the developing controller. However, this is not the only way an observer may view neural networks. Some networks which do not allow changes to neuron and synapse properties might be viewed as developmental with, for example, some parts of the network instantiating the behaving controller and other parts the developing controller.

In this thesis the two tiers ('controllers') are implemented explicitly as separate systems, which allows for simple study of the phenomena observed. The behaving controller is expected to be formed by the interaction of existing neurons and synapses, and the developing controller is expected to consist of changes to synapses and growth and death of neurons and synapses. As will be seen later, in practice, the separation of these levels is not always so simple.

This definition of development includes the processes of morphogenesis, growth and ageing, and learning. However, these processes may have different properties. Whereas morphogenesis may sometimes involve changing from a simple or random state into a complex, ordered state, the others, especially learning, involve changing from one complex, ordered state into another. Most, but not all, of the work detailed in this thesis involves processes like morphogenesis, moving from a simple or random state towards a complex, ordered one.

It has been shown that neural networks with fixed structure and leaky integrator neurons (CTRNNs, see section 5.2) are capable of learning behaviour (Tuci et al., 2002b). CTRNNs have been shown to be capable in principle (if they are large enough and have wide enough time constant ranges) of recreating any given smooth dynamical system (Funahashi and Nakamura, 1993). This includes developmental systems, but there are advantages in modelling separate developmental processes explicitly, and in practice evolving developmental systems in CTRNNs may be very

much harder than this theoretical result might initially suggest.

Much current evolutionary robotics work uses CTRNN controllers. The work discussed here aims to enhance the clarity with which developmental systems may be studied by enforcing explicit separation between tiers. By explicitly modelling developmental processes separately from the behavioural processes of a robot's controller (and complementarily restricting the behavioural processes to operate over relatively short timescales), the advantage is gained that the process of development may easily be observed separately from that of behaviour.

1.2.2 Development is not evolution

Some research treats evolution as being almost equivalent to development. For example, Ackley and Littman's (1991) experiments into the Baldwin effect modelled learning as a temporary phase of random mutation. While this is acceptable as a greatly-simplified model, it is dangerous to expect development to act on the same systems as evolution. While evolution modifies the genetic material of related individuals over generations, development is the process of change within the phenotype.

For example, evolution might modify the number of fingers which a person develops, but development will consist of the growing in size of the hand during childhood. To confuse developmental changes with evolutionary ones here would mean suggesting that more fingers could be grown in response to a particular environmental stimulus: in practice, this type of development is not observed under normal conditions. Evolution may often be seen as providing the capacity for certain types of development to occur, but the developmental process itself is very different.

For some time, it was believed that the development of an individual followed its species' evolutionary history, but this theory has been thoroughly discredited (Gould, 1977).

More subtly, many experiments attempt to shed light on animal brains (which have been produced by development and learning) by 'teaching' neural networks through an evolutionary process to do some cognitive task (Beer, 2000; Dale, 2002). This approach is not without value, since artificial evolution has some characteristics in common with developmental processes - behaviour may be produced without specifying the exact process of how it is generated. However, it is important to realise that development and evolution are very different processes; not only do they operate over hugely different timescales, but they are of very different natures. Using evolution to model development may be misleading if caution is not exercised. The work described here models development as a completely different and separate process from evolution over which evolution has only indirect control. For more on the viewpoint that the differences between evolution and learning are significant, see (Nolfi, 2002).

1.2.3 Note on the word 'plastic'

For the purposes of this thesis, the word 'plastic' is only used in the context of 'plastic neural networks,' which should be considered an abbreviation for 'neural networks whose synaptic weights are subject to plastic change.' No implication should be taken that such controllers are in any sense more plastic (more capable, in general, of plastic change) than any other controllers.

1.3 Summary

This work serves two goals: development of tools for studying development, and the use of those tools to investigate some open questions. Since the investigations presented are not rigorously controlled they represent exploratory experimentation only, and the results found, whether positive or negative, are not to be relied upon as supporting any general hypotheses.

1.3.1 Development of tools

As in any relatively new area of study, there is a need in the area of developmental evolutionary robotics for a good set of tools which may be used to shed light on the matters which are of interest. This thesis discusses in detail the motivations, design and practical working of several tools, which it is hoped may be useful in future study. The most important of these tools are the two developmental controllers which were designed.

The neural network controllers described in chapters 6 and 8 were designed with an explicit separation between the behaving part of the controller and the developmental part. This separation, although in practice not complete, allows the experimenter to study the phenomena of interest in relative isolation: explanations of how the agent's behaviour is generated may begin with the assumption that 'development' and 'behaviour' take place in the 'developmental' and 'behavioural' parts of the controllers dynamics, although some refinement of this view may be needed in due course.

The controllers presented here exhibit several unique properties: the controllers of part 1 combine properties such as spatial separation and chemical gradients, influence of experience on development, a single genotype for all units and behaviours of cognitive interest, some of which have been used in previous work, but which may not have been combined together into a coherent whole before. Similarly, the controllers of part 2 represent a practical scheme for evolving developmental controllers which may be successfully evolved to perform many different tasks - again the application of a single developmental controller type to so many differing and varied tasks may not have been undertaken before.

Both controllers have been shown to be practically useful in that they may be evolved to perform simple tasks, while at the same time they exhibit behaviours which may be of interest for those studying development. Furthermore, through the work described in the next section, these controllers have been shown to be useful tools in the exploration of open questions in the area of development and its interactions with behaviour and evolution.

In addition to the controllers themselves, the methods used for evolving and studying the evolution and behaviour of these controllers in many different scenarios may prove to be useful additions to the developmental researcher's methodological toolkit. Many lessons have been learned about how to evolve developmental systems (and how not to evolve them), and these are presented throughout the thesis, most notably in sections 7.3, 8.1 and chapter 10.

The most important lessons learned involve providing structure for evolution to work with (rather than an empty system to be grown from nothing), using fitness functions that prevent suboptimal non-developmental methods from being adopted, and understanding and controlling the extra degree of complexity that is introduced with the adoption of developmental systems - other factors may need to be simplified to allow feasible study.

All of the tools described were developed in the pursuit of the practical goal of learning about development through the exploration of open questions. This work is described in the next section.

1.3.2 Exploring open questions

Throughout this work the goal has been to explore open questions about development - to shed some light on some of the many speculations that have been made about developmental systems. These open questions are outlined in chapter 4, but they are outlined briefly, along with the results of the investigations, here. Attempted explanations for the results outlined below are made in chapter 10.

An over-arching idea has been that developmental systems may be designed which evolve successfully to perform the kinds of tasks that are performed in evolutionary robotics by non-developmental controllers. Some evidence to support this idea has been found, but it seems clear from both part 1 and part 2 that while it is possible to evolve developmental controllers to perform these tasks, it is certainly not as easy as evolving similar non-developmental ones.

A second open question about development is that a developmental controller will be more able to evolve to undergo a predictable change in the lifetime of the agent, than a non-developmental one. This question is based on the idea that an agent which is able to change the structure of its behaving controller might be able to use that ability to produce different behaviours at different times, whereas a more static controller would need to produce different behaviours using the same controller structure. In the investigations of this question, unexpectedly, the non-developmental controllers performed better at a task involving predictable change than the developmental ones.

A further question on the subject of predictable change is that agents evolved under selection pressure to undergo change in their controllers at the time at which a predictable change in behaviour is required, would evolve to perform that predictable change more successfully. The idea behind this question is that if selection pressure is used to push agents down an evolutionary path involving change in the controller at the time when a change in behaviour is required, the already-existing controller change may be harnessed later in the evolutionary process to produce behavioural change. In the investigations of this question, agents evolved with this extra selection pressure in a task involving predictable change performed significantly worse than those evolved without this selection pressure, which was not the expected outcome.

A fourth open question concerns whether developmental controllers will be capable of performing more complex behaviours than non-developmental ones. This question is inspired by the idea that in order to produce more complex behaviours, more complex brain structures are required, and it may be easier to produce useful complex structures under evolution using a developmental system (for example a system which exhibits self-organisation) than using a non-developmental system. This question was not strengthened through this work, since the developmental controllers generally performed worse at the tasks for which they were evolved than the non-developmental ones. However, the developmental controllers performed better as the complexity of the tasks increased, which suggests that further study may be justified into whether they might outperform non-developmental controllers in tasks of sufficient complexity.

A fifth open question is whether developmental controllers may be more easily evolved to perform tasks involving learning than non-developmental ones. Similarly to the previous point,

the basis for this idea is the idea that if a developmental controller can undergo structural change in its behaving controller, that structural change could be used to adapt the behaving controller in response to learning, which may be easier to evolve than a controller which adapts its responses without undergoing structural change. In the investigations of this question, the developmental controllers performed better than the non-developmental controllers in learning tasks, but since the developmental controllers were capable of change over longer timescales and had more free parameters, further investigation is required.

A sixth open question is whether developmental controllers may evolve more easily to perform different behaviours when confronted with different environments. The idea behind this question is that a developmental process may be able to use environmental triggers to produce entirely different controllers in different environments, whereas a non-developmental controller would be required to produce different behaviours in different environments with the same controller structure. In the investigations of this question, the developmental controllers did not perform significantly better than the non-developmental ones in a task requiring different behaviour in each of two different environments.

A seventh open question is whether evolved developmental controllers may be more capable of adapting successfully to overcome changes in the agent or environment which were not present during evolution than non-developmental controllers. The idea behind this question is that since a developmental controller must 'construct' its behaving controller in every lifetime, and this construction may be influenced by environmental factors, if the environment is changed, the construction process may also be changed - possibly in ways that preserve the original behaviour of the agent. The controller containing explicit processes of development scored more highly in some tests of robustness than a more standard neural network controller with temporal dynamics which were allowed to vary over the full lifetime of the agent, and which had a similar number of free parameters. Thus it appears that in some tests of robustness the explicit developmental process introduced facilitated robustness more than the simple addition of widely varying timescales and large numbers of free parameters.

An eighth open question is whether agents evolved to develop under noisy conditions within their controllers would be more robust than those evolved under more predictable conditions. The idea behind this question is that if the self-construction of the agent is generally executed under varying conditions, the construction process may become evolved to resist other variations, such as changes in the body or environment of the agent. In the investigations of this question, the performance of agents evolved with noisy developmental processes was consistently lower than that of those evolved under less noisy conditions both during normal evolution and within the robustness tests, providing no support for the ideas behind the question.

A ninth open question comes out of the work described in section 3.3. The suggestion is that controllers whose development consists mainly of synapse death will be easier to evolve than those whose development consists mainly of synapse growth. This belief (extrapolated from the view held by the 'selectionists') is motivated by the idea that brain growth may be Darwinian in nature, with competition between synapses (resulting in their death in some cases) producing useful structures, while the growth of synapses is largely random or uniform, not guided by the experience of the animal. The investigations into this question were inconclusive, but if anything

suggested that the opposing view (that of the 'constructivists') may be correct, in that controllers whose development consisted mainly of growth outperformed those whose development consisted mainly of death in some experiments, while in others the performance was similar in both cases.

1.4 Contributions of the thesis

This thesis contributes to the field in the following two areas: development of experimental tools and investigation of open questions. The former is presented as the major contribution of the work, while the latter demonstrates the utility of the tools developed in practical experimental work.

1.4.1 Tools development

- Two new classes of neural network controller have been developed which may have utility in the study of developmental dynamics. These controllers combine novel combinations of features inspired by natural systems and have proven effectiveness in practice, having been successfully evolved to perform a number of simple tasks.
- Practical issues and difficulties in the artificial evolution of developmental systems have been brought to light, and ways to overcome some of them outlined. Solutions include the provision of appropriate structure and the use of specialised fitness functions.
- Several conceptual issues, including the importance of symmetry and the need for pre-existing structure through which the developmental process can operate, have been discovered and explained.

1.4.2 Investigating open questions

- Open questions about the potential advantages (greater flexibility, adaptability, robustness and ability to produce more complex behaviour) of using an explicit development process in robot controllers have been investigated. In several cases little or no evidence has been found to support the ideas behind these questions. Possible explanations for these results have been offered.
- Open questions about ways of influencing the evolutionary process to enable the evolution of higher-fitness developmental controllers have been explored. These involve selection for change in the controller at the time at which behavioural change was required, and the introduction of noise into the developmental process to increase robustness. In both cases, against the expectations of the author, these mechanisms were not found to have the positive effect predicted in the conditions tested.
- One of the controllers developed, the 'growth and pruning' controller, has been shown to be more robust in some cases than controllers without explicit developmental mechanisms (but with comparable size in terms of number of genetically-controlled parameters and comparable temporal ranges of behaviour) to certain disruptions of a robot's body and environment.
- A comparison between different levels of growth and removal of structure in controllers has been undertaken whose results may have relevance to the debate in neuroscience between constructivist and selectionist models of brain development. It has been shown that guided growth appears to be more useful in these developmental systems than guided death, potentially providing weak support for the constructivist viewpoint.

Chapter 2

Background in evolutionary robotics

2.1 The overall research programme

Evolutionary robotics, and the wider field of artificial life, study artificial systems which have, or appear to have, similarities to natural systems. By studying these artificial systems it may be possible to investigate the dynamical character of lifelike systems and develop conceptual, statistical and practical tools which help to classify and understand them.

This programme has properties in common with thought experiments: the process begins with a set of assumptions which are then programmed into an artificial system, and the outcome investigated. However, in practice the complexity of the systems means that the work resembles empirical science since analytical solutions are rare, and the systems are most often understood by means of a trial and error style similar to empirical investigation. Di Paolo et al. (2000) coined the phrase 'opaque thought experiment' to explain this scenario: a thought experiment is being performed, but the consequences of the assumptions are not immediately obvious (they are 'opaque') and thus investigation is required to find them. The work described in this thesis fits into the above description, being intended to shed light on the types of mechanisms that produce interesting behaviour.

Much of the work in evolutionary robotics studies the properties of different types of controller evolved to perform certain tasks. Often the controllers used are specified in genotypes, and change over evolutionary timescales (from generation to generation), but do not exhibit much change during the lifetime of a particular individual. The work here represents an attempt to extend this work to capture some of the different dynamics involved when structural change to a controller is allowed (or required) during a robot's lifetime.

For a book providing a definitive overview of evolutionary robotics, see (Nolfi and Floreano, 2000).

2.2 Previous work

The work on developing neural networks falls roughly into two categories: experience-independent and experience-dependent. The next two sections cover work in each of these areas.

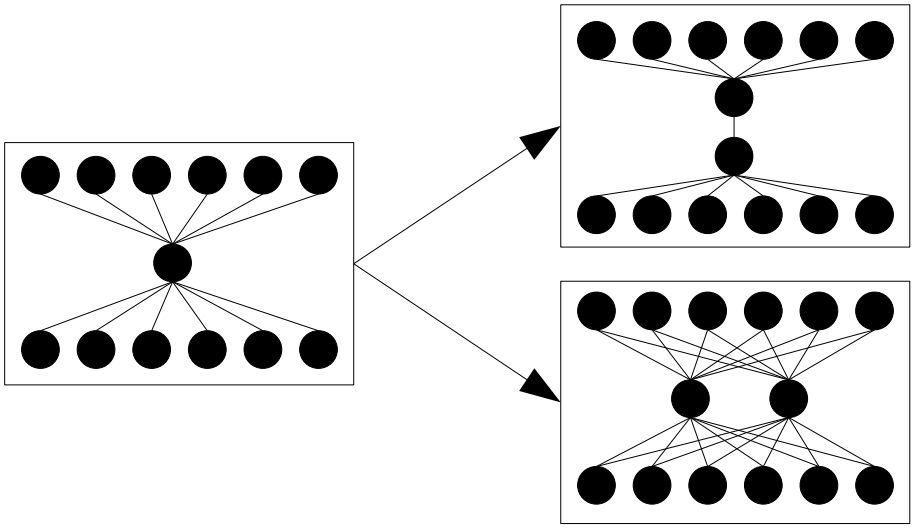

Figure 2.1: Some of the genetically-defined operations which are applied to the growing neural networks in Gruau's (1995) work. The network on the left may be transformed into one of the ones on the right, depending on the instruction being interpreted in the genotype. There are several other types of possible operation, including changes in weight values and neuron copy operations which only copy some of the connecting synapses. This figure is a partial copy of one found in (Gruau, 1995).

For further coverage, there are good overviews of the literature available on development in evolutionary robotics models, in book chapters by Nolfi (2002) and Cangelosi et al. (2003).

2.2.1 Experience-independent change

The attraction for some researchers of using developmental mechanisms has been the possibility of the construction of large neural networks using relatively small genotypes. This is often achieved by explicitly implementing modularity in the growth system, so that parts of the genotype may be used several times to build different parts of the controller. In this case development is used simply as an indirect mapping from genotype to phenotype, and this goes on before the agent's life begins, independently from its experience.

Gruau (1994; 1995, Gruau and Whitley (1993)) used a tree-like genotype with a graph-rewriting grammar to create modular neural networks. The system, which involved each section of the genotype being interpreted by the neuron to which it applied rather like a computer program with instructions such as 'divide into two cells and assign branch A of the genotype to the first cell and branch B to the second,' was successfully used to evolve locomotion in a simulated six-legged agent and other tasks. Some of the types of operations that go on during the development process are illustrated in figure 2.1.

Gruau proved the concept was feasible by hand-designing a solution to the walking task using his system, and then showed that such a solution may be evolved. While the hand-designed solution used six identical subnetworks, one for each leg, the evolved solution built a single subnetwork for each pair of legs, which was repeated three times. These solutions met the aim of the experiment which was to show that the 'cellular' encoding scheme enables evolution to re-use

parts of the genotype to construct networks consisting of repeated structures. It did not address any of the other issues mentioned in section 4.1.3 since all development took place before the agent's lifetime began.

Other work using string and graph rewriting to develop neural networks may be found (Belew, 1992; Husbands et al., 1994; Kodjabachian and Meyer, 1998; Vaario, 1991).

Jakobi (1995) designed a developmental neural network scheme modelled on the biological mechanisms of neuron growth. It involved models of DNA molecules, protein transcription and diffusion as well as a genetic regulatory network. The controller was seeded 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. Controllers were successfully evolved to allow agents (simulated Khepera robots) to perform two behaviours: corridor following and obstacle avoidance. However, further attempts to evolve more complex behaviours proved unproductive. It is likely that this failure may have been connected with the high level of complexity involved in the developmental process (some of which is illustrated in figure 2.2), involving genome preprocessing, a genomic regulatory network, protein transcription, different protein classes, and a two stage development process producing a genomic regulatory network which in turn produced the neuron properties and connectivity. The complexity of this model was an influence on the work in this thesis: it was decided that by adding only the minimal level of complexity to a standard model, both evolution and understanding would be aided.

Dellaert and Beer (1994b; 1994a; 1996, Dellaert (1995)) performed simultaneous evolution of the morphology and controller of agents with a developmental system that consisted of a genetic regulatory network controlling the division of cells that formed a grid containing sensors, motors and neurons. They constructed two systems, one of which was relatively closely modelled on biological development, and turned out to be difficult to evolve to perform simple behaviours. The other system was simpler, making it possible to evolve a line-following behaviour. (For more work involving genetic regulatory networks, see (Eggenberger, 1997).)

Cangelosi et al. (1994) used a spatial system of cell division and migration to develop controllers capable of navigating to different locations in a grid world. Some stages of the development of the controllers are illustrated in figure 2.3.

There are other examples of experiments in experience-independent development of neural networks, including those of Nolfi and Parisi (1992) whose development process sometimes continued during an agent's lifetime, but remained independent of its experience. Other examples may also be found (Astor and Adami, 1998; Belew, 1992; Fleischer and Barr, 1993; Goodwin, 1994; Harp et al., 1989; Kelso, 1995; Kitano, 1994; Lindenmayer and Rozenberg, 1976; Wilson, 1987).

2.2.2 Experience-dependent change

There have been several pieces of work that use developmental processes to construct neural networks whose structure and properties may be influenced by the experience and behaviour of the agent whose behaviour they control.

Nolfi et al. (1994; 1996) used a developmental neural network system to show that agents can be evolved that behave differently under different circumstances using the same genotype. The

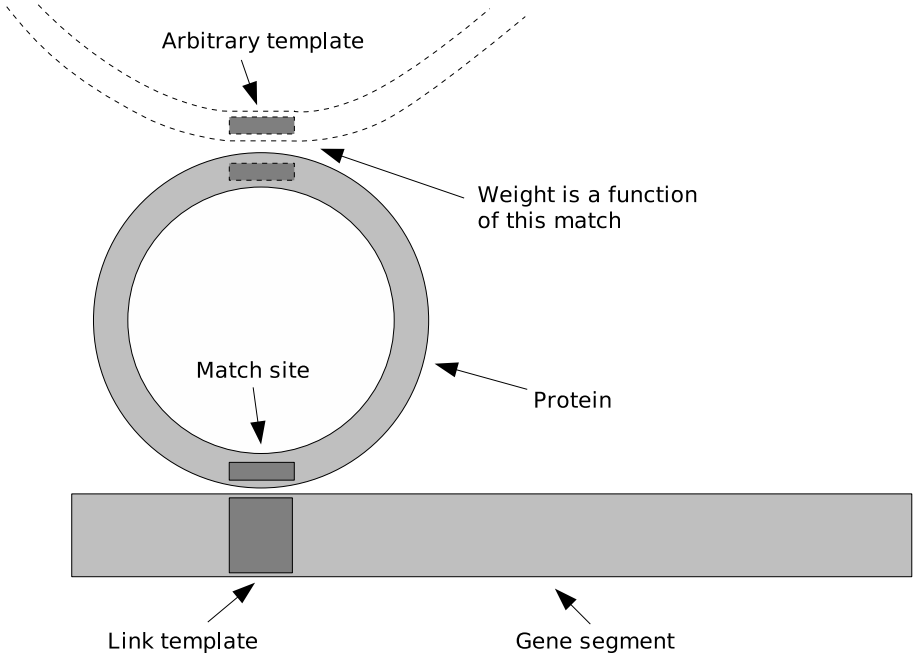

Figure 2.2: The protein matching stage of the model described in Jakobi (1995). Proteins are encoded in the genotype as strings of characters formed into a circle. Matches between existing proteins and genome segments cause the activation of certain genes, which in turn produce proteins. The resulting genomic regulatory network, consisting of different classes of protein, is found in a preprocessing stage before being used to grow the neural network with different protein classes triggering events such as cell division and movement. This figure is a copy of one found in (Jakobi, 1995).

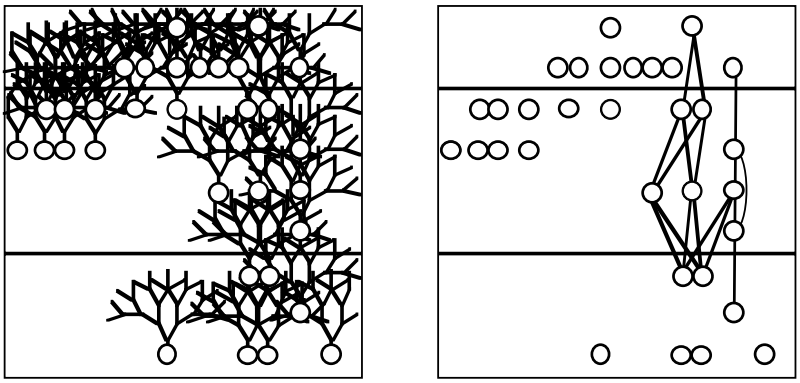

Figure 2.3: The last stages of development in the controllers used by Cangelosi et al. (1994). In the figure on the left, the cells have divided and migrated, and the axonal growth phase is complete. On the right is shown the resulting neural network. Neurons near the bottom of the controller are linked to sensors, and those near the top are linked to motors. This figure is a partial reproduction of one found in (Cangelosi et al., 1994).

developmental system is based on back-propagation with the training signal being provided by the agent itself. These systems allow the experience of the world to affect developmental outcomes for an agent operating in a grid world, and for a robot navigating in an environment which varied over different generations between being dark or light.

These experiments emphasise the need for agents to be able to influence their own experience and thus change the path of the developmental process by their behaviour. The need for agents to react differently in dark and light environments results in a situation similar to that described in this thesis in section 8.2.11, where an agent is required to respond differently to identical stimuli when other aspects of the environment are different.

Nolfi and Parisi were able to show that networks that develop using a back-propagation learning system with a training signal provided by the agent itself were better able to explore the light and dark environments by using the learning mechanism to behave differently in the two different environments (this is illustrated in figure 2.4). Networks without the learning mechanism used a single strategy for both types of environment and thus performed less well. Since preventing the learning networks from undergoing learning resulted in performance worse than that of the non-learning networks they concluded that the evolved initial connection weights were selected for a predisposition to learn rather than for optimal immediate task performance.

This thesis shares many of the same motivations and guiding principles as the work described above, and explores many of the same issues, attempting to extend these ideas to more general developmental mechanisms and more complex tasks.

Other examples of the use of development to construct neural networks are available (Kitano, 1994; Nolfi and Parisi, 1995; Cangelosi, 1999).

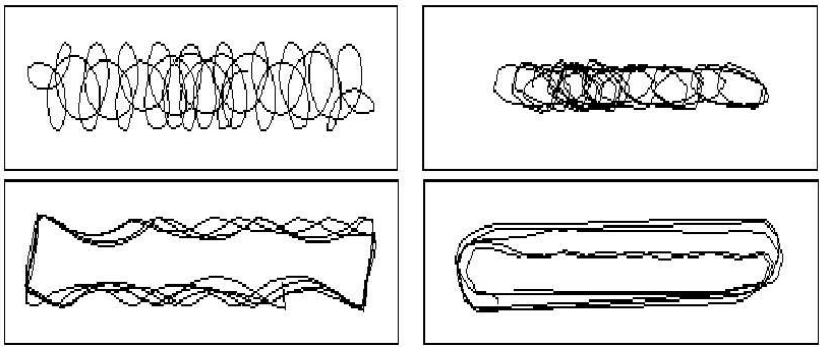

Figure 2.4: The paths taken by two evolved individuals from Nolfi and Parisi's work on agents that are able to adapt to different environments. The non-learning individuals are shown in the top row and the learning individuals are in the bottom row. Behaviour in the dark environment is shown on the left and in the light environment on the right. The learning agents are able to adapt to the differences in the environment and behave similarly in the two environments whereas the non-learning agents behave differently depending on what type of environment they are born into. This figure is reproduced from (Nolfi and Parisi, 1996).

Much other work on structural change affected by the agent's experience uses plastic neural networks (Floreano and Mondada, 1996, 1998). These controllers have Hebbian-like learning rules that apply to each synaptic connection, changing its weight according to the activity of the pre- and postsynaptic neuron firing rates. Floreano and Mondada used four different learning rules to allow the connection weights to change during the lifetime of the agent, and did not specify initial weight values, so that the weights were determined simply by the outcomes of the application of the learning rules. A key finding was that the resultant controllers did not simply rely on static weight values being approached, but used dynamic weight changes as part of the control system. The plastic controllers used in part two of this thesis are based on these controllers, but with a single learning rule in place of the four different ones used here.

Plastic neural network controllers have been shown capable of performing very simple tasks such as wall-following and obstacle avoidance (Floreano and Mondada, 1996, 1998; Elliot and Shadbolt, 2001). They have also been been used to investigate learning (Tuci et al., 2002b; Fernando, 2002) and homeostatic change (Di Paolo, 2000) and shown to be capable of evolving relatively very complex behaviours.

Recent work has investigated spike-timing dependent plasticity to alter the synaptic weights in a network of spiking neurons (Di Paolo, 2003; Kempter et al., 1999; Rao and Sejnowski, 2001).

The work described in this thesis attempts to go beyond some of the research described above by allowing for large structural changes to occur (including death and growth of neurons and synapses) within a system that has all the additional advantages (both practical and theoretical) of being affected by the experience and behaviour of agents embodied and situated within complex dynamic environments.

Further influences on this work come from the biological sciences. Some of these are described in the next chapter.

Chapter 3

Background in biology

Both theoretical and experimental biology provide concepts, frameworks and experimental setups that influence this work. This chapter describes some experimental evidence for the need to understand development in order to understand learning and cognition and then outlines some of the theoretical ideas underpinning the research programme. Finally, it sketches the parameters of a specific ongoing debate to which this work may contribute.

3.1 Development's role in learning and cognition

Work in neuroscience has recently 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. (2002) 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, displaying action potentials and synaptic inputs similar to those found in mature cells. The fact that these structural changes in real brains, including growth of new neurons, may be important to the functioning of those brains (since the new neurons are incorporated functionally) suggests that there is real value in studying similar structural changes in artificial systems, in the hope of understanding how the behaviour of those systems is affected by the capacity for such changes.

Meanwhile, momentum for the general view that change is vital and pervasive in adult brains is continuing to grow (for a review of some of the work in this area see (Kolb et al., 1998)). Ivanco and Greenough (2000) 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. They show that such structural change goes on throughout the lifetime and is important

for brain function. This work too provides strong motivation for the work in this thesis, since it implies that the kinds of adaptation and repair which are of interest to robot designers may be brought about in the brain through structural change.

Elman (1993) argues that the long maturation phase in human development is crucial to facilitate the large amount of learning that takes place during that period. He uses neural networks being trained to process sentences in an artificial language to illustrate his point: he shows that an unconstrained network does not learn successfully, but if the network's memory is constrained to be short at the beginning of the training and is lengthened over time, it is able to learn the required language structures. He argues that this success shows a possible explanation for the posited critical period in human language acquisition, during which children are more capable of learning language than at later times. The explanation is that the learning that takes place when the network is constrained simplifies the weight space to be searched, limiting it to useful areas and preventing the network from moving into incorrect local maxima. This work emphasises the need to consider development in any study of learning.

The important of experience-dependent structural change is emphasised by the work of Purves (1994), who showed that there is evidence that neural connectivity is affected by the activity of neurons. This observation is crucial to the design of the experiments in this thesis, since here development is seen not as a kind of information compression system or pattern generator, but as a way of absorbing environmental influence into the controller structure and adapting to changes in that environment.

For an overview of the work suggesting the need to understand structural change in brains in order to address questions about cognition, and the idea that neural structures are constructed in a complex interaction between intrinsic and experience-dependent processes, see (Quartz and Sejnowski, 1997; Quartz, 1999). An attempt to synthesise the conflicting viewpoints of those who suggest that many cognitive skills are 'hard-wired' as innate skills and those who argue that almost nothing is innate but rather cognitive structures are constructed through interaction with the environment is given by Karmiloff-Smith (1992).

As the evidence begins to point to the idea that multi-tiered processes of structural change are pervasive in the cognitive and behavioural systems of animals, the need to understand such processes becomes clear. This thesis attempts to work towards this understanding by generating and studying processes which are in some sense analogous to these natural developmental systems. The analogy is inevitably extremely weak, but through such poor imitations it is hoped that enough information will be acquired to generate better models and progress over time towards a more complete understanding.

3.2 Development and evolution

3.2.1 Canalisation and genetic assimilation

The role that development plays in evolving populations is not fully understood. However, the pioneering work of Waddington (1975) helped open many questions for investigation.

Waddington saw development sometimes as a stabilising force, acting in opposition to evolution in some circumstances, through the process of canalisation. This process means that similar phenotypic outcomes occur even given differences in the genotype and the environment. Any

study of development must be a study of the canalised pathways formed through evolution to construct viable outcomes.

This view of developmental balancing evolution (or at least mutation) has not been reflected in much of the evolutionary robotics literature: often here development is seen as a way of generating more complex outcomes from simple starting points, or as a way of producing repeated structure, but not as a stabilising force. In nature, stable developmental pathways are needed to force a system which in principle could be very sensitive to initial conditions down a consistent pathway.

A developmental system which resulted in wildly different outcomes each time would not be open to evolution since its fitness would not be consistent, and so in nature this type of developmental system does not arise. In artificial systems, however, it is quite possible that a very unstable developmental system could be designed. Thus it is a prerequisite for any developmental system which will exhibit evolvability to allow for some form of canalisation to occur, preventing instability of outcomes in the face of environmental noise and genetic variation.

Waddington's description of genetic assimilation demonstrated a mechanism whereby 'acquired' characteristics could become incorporated to such an extent into the developmental process that they became 'inherited,' producing a process that appeared to be similar to Lamarckian evolution (Lamarck believed that acquired characteristics were inherited directly from the parent). This idea provides evidence for the arguments of Oyama that such terms are effectively meaningless (see next section). Genetic assimilation is one mechanism by which developmental systems could offer an evolutionary advantage over non-developmental ones. If useful structural changes are first allowed by evolution under the right environmental circumstances, they may later be enforced by evolution to be present at all times.

3.2.2 Acquired and inherited characteristics

Oyama (2000) argues that while most researchers claim to avoid the nature-nurture distinction, in practice they often continue to apply it. She argues for a new approach of 'constructive interactionism' which sees every organism and interaction as a product of the evolutionary and developmental historical processes leading up to it, none of which may meaningfully be divided into inherited and acquired, or genetic and non-genetic. This requires a huge paradigm shift, especially in the world of evolutionary robotics where genotype, development and behaviour are often explicitly defined and separated.

It is interesting to note that these arguments still apply in situated and embodied robots: the distinction imposed by the experimenter on such systems between behaviours or features 'caused' by the genotype and those 'caused' by the environment is still meaningless since in the presence of a different environment the 'genetic' behaviours are unlikely to happen, and equally in the presence of a different genotype the 'environmental' behaviours would change. In evolutionary robotics we often see that some features of an agent are identical no matter what environment it is presented with (unless the properties of the agent itself are changed), and in this case we may properly regard them as being caused by the genotype. However, the interesting features (often the behaviour of the robot) are always caused by a complex interaction between agent and environment.

Similar ideas are explored by Elman et al. (1996) who argue that genes may be seen as cata-

lysts for development, rather than blueprints for morphologies, and that these catalysts may work at many different levels from molecular to behavioural. They emphasise the importance of regulatory genes that control the timing of events, which is a theme encountered in the developmental systems studied in this thesis. They argue against the idea that knowledge of innate abilities such as universal grammars are contained in the genome, but instead claim that such a concept is meaningless since knowledge may only be understood in terms of brain structures, and that such knowledge may only be acquired through developmental processes. They claim that since the connectionist models they describe are capable of performing cognitively interesting tasks such as linguistic processing, the idea that such tasks may only be performed by fully-specified innate systems is disproved.

By introducing development into artificial systems, we move towards a situation closer to that found in nature, where no feature or behaviour may properly be explained without reference to the complex interactions between genotype and environment, or, more accurately, the historical processes of evolution and development leading up to the currently active process.

If inherited and acquired characteristics are actually identical, or located on a single continuum, it follows that in order to learn about adaptation and learning, the acquisition of such characteristics through development must be included within the scope of our investigations. Agents must be regarded as processes (in fact sub-processes of the evolutionary process) rather than static products of evolution.

The work described in this thesis, like most of the current work in evolutionary robotics, makes explicit the influences of genetic and non-genetic factors on the some of the structures within the agents. Nevertheless, the argument that the outcome is a product neither of the one nor the other, but of the whole, still holds. Behaviours and neural structures such as those found in section 7.2 and 9.2, require the integration of both genetic and other factors for a full explanation of their origins.

3.3 A live debate: constructivism and selectionism

A debate is going on within theoretical neuroscience between two groups of researchers with opposing views on the nature of the process going on during brain development. This debate relates strongly to a theme of this thesis about the need for refinement of pre-existing structure rather than directed building of structure from sparse networks.

Purves et al. (1996) frame the debate in an article entitled ‘Is neural development Darwinian?’ in which they claim that some tacit assumptions are being held by many neuroscientists against the available evidence. The assumption is that Darwinian-like processes select neuronal elements (neurons, neuronal branches, synaptic connections and groups of interconnected neurons) on the basis of whether they produce adaptive behaviour. Purves et al. claim this assumption is widespread but unquestioned, and furthermore that it is often simplified to mean simply an initial overproduction of material which is pruned away later.

They argue that the evidence for overproduction of neural elements is sparse in many areas and is flatly contradicted in others: brain mass increases over time, and since the number of neurons remains approximately constant they argue that the nerve cells and the complexity of their arborisations become larger over time, rather than being pruned into greater simplicity. A similar

viewpoint is presented by Quartz and Sejnowski (1997).

The counter-argument comes from Sporns (1997), and Changeux (1997) who argue that Darwinian processes need not result in a reduction in neuronal elements from an initial excess since the processes of variation and selection may occur simultaneously, rather than being temporally separated. Furthermore, they criticise the alternative concept put forward, 'directed growth' of neuronal elements, pointing to evidence suggesting that change comes about through selection from many small undirected variations.

To some extent both sides in the debate appear to be answering straw man arguments from the other side - on one hand, the selectionists are not claiming that an explicit two-stage process of overproduction and then pruning takes place, but rather than this process is ongoing, with continuous unguided growth and competitive death or weakening of synapses occurring simultaneously, and on the other hand the constructivists are not claiming that such competitive processes do not go on, but are simply pointing out some of the evidence that this may not be the only way that genetic and environmental influence combine to shape brain structure.

Nevertheless, this debate raises some interesting questions, and some of them may be informed by the work in this thesis, since one of its over-riding themes is the need for initial structure with which development can work in order to produce the necessary useful structures. Some of the experiments in part two are designed in an attempt to shed some light on the situation from an evolutionary robotics perspective, by looking at the question of whether growth or pruning of structure is a more useful way of generating behaviour-producing structures.

The next chapter collects together the motivations for this work, and the open questions to be explored and lays them out explicitly, to be addressed in the following chapters.

Chapter 4

Motivation and Open Questions

4.1 Motivation - why study development?

Development has been overlooked by some evolutionary robotics researchers: sometimes in the past the computing resources available were too limited to model developmental processes in satisfactory ways, but more significantly some people have seen development as fairly irrelevant: bodies are seen as being built by genes, and the specifics beyond that are treated as unimportant, since it is assumed that evolution is powerful enough to work within any system to produce highly adaptive products. However, recently it has been becoming increasingly clear that the specific properties of a system crucially affect the structures that arise as the result of an evolutionary process. The fact that behaving animals are not static systems but developing ones must be taken into account if systems with similar properties are to be generated.

Many researchers believe that modelling development will bring advantages in two areas: natural science, where the expectation is that by generating examples of artificial developmental systems it may be possible may learn useful information about development that may inform our study of natural organisms, and engineering, where the expectation is that development will assist in the development of agents capable of performing more complex or more difficult behaviours than those currently in existence.

By removing one aspect of 'hand-crafting' from the evolutionary robotics process, since the experimenter makes fewer arbitrary decisions about the experiment (such as the number of neurons and synapses), advantages may be gained from both the scientific and engineering perspectives. Every experimenter wishes to reduce the number of assumptions made in order to improve the chances that the results may generalise, and so development may be attractive from this viewpoint, but similarly, an engineer may well be happier putting such choices under evolutionary control, rather than using a trial-and-error approach to find good parameters.

Sections 4.1.2 and 4.1.3 discuss two motivations for the study of development, and the next section addresses the question of whether CTRNN controllers are sufficient for the study of developmental systems.

4.1.1 Are CTRNNs developmental enough?

It has been argued (Harvey, 2005) that since they contain internal state and have been shown to be universal approximators (Funahashi and Nakamura, 1993) of smooth dynamical systems, continuous-time recurrent neural networks (see section 5.2) are sufficiently flexible that the addition of a further set of dynamics is superfluous. There are two counter-arguments to this proposition.

First, the property of universal approximation is proven as a theoretical result, but its application is limited in practice since no comprehensive information is available on whether it is possible to evolve these networks to display a given set of dynamics in a feasible time. Anyone who has worked with evolving CTRNNs knows that it is far from guaranteed that the desired dynamics are successfully evolved in any particular situation. Some types of dynamics are easier to evolve than others with CTRNNs, and the intrinsic properties of the neurons make some dynamics require large numbers of neurons to produce. Many experiments limit the number of neurons available to evolution, effectively preventing the expression of certain types of dynamics, since an unlimited number of neurons is required for the universal approximation result to be valid.

Second, it is sometimes argued that the particular dynamics produced by CTRNN models resembles the dynamics of networks of neurons. If this description of CTRNNs is accepted then it is certainly reasonable to introduce another process whose dynamics is designed to resemble that of development.

Evidence is growing that CTRNNs may need to be extended to allow the evolution of more sophisticated behaviours. It has been shown that learning behaviour is difficult (but not impossible) to evolve in CTRNNs (Tuci et al., 2002b), and that path integration behaviour is much easier to evolve in models extending the CTRNN system (Vickerstaff and Di Paolo, 2005) than in ordinary CTRNNs.

For the purposes of this work, however, one does not need to rely on these arguments to justify the use of an extended CTRNN model since the explicit separation between two levels of operation in the controllers is inherently useful in order to understand how such two-tiered systems evolve and behave in a situation where there is no confusion about the level at which any given phenomenon is acting.

4.1.2 To learn about how it works

There are rich possibilities for the study of models of development to feed into biological and psychological studies of development, cognition and behaviour. By making development a separate process from evolution the analogy could be improved between the natural systems to be understood and the artificial ones being generated and studied. This would mean that phenomena observed in artificial systems were more likely to lead us to insights into natural systems, and questions from nature could be explored in more relevant simulated scenarios.

In the natural world, development is a major constraint on the paths taken by evolution. Understanding the interactions between behaviour, development and evolution is essential if progress is to be made in understanding each of these systems.

The naive view of phenotypes being generated from genetic ‘blueprints’ has been shown to be incorrect and unhelpful by Oyama (2000). She argues that the developmental history of an organ-

ism is crucial to its character, and that any distinction that is made between innate and acquired characteristics is always a false one since they are inseparable. If these observations are taken on board, systems must be modelled with dynamics analogous to development if living systems are to be understood, since the dynamics of development are core to their structure.

By modelling development, opaque thought experiments may be performed comparing, for example, two different theories about brain development, and conclusions drawn about which system is more likely to evolve, and which is capable of producing useful behaviours. As research continues, an understanding may be built of what kinds of dynamics produce useful and lifelike behaviours, and this understanding may feed into the types of dynamics being searched for in natural systems.

Of course, this is a very ambitious and speculative research programme which is in its most basic first stages. It is always difficult to justify a suggested similarity between artificial and natural systems, and there is a great deal to be learnt before truly valid analogues may be found (and proved valid). At present it is simply hoped that very general conclusions may be drawn about the kinds of system that may produce certain classes of outcome.

This section and the preceding chapters touch on many different assumptions, theories and questions about development. In order to increase our understanding of this area, these ideas must be clearly stated, and explored through experiments. Section 4.2 attempts to make explicit the open questions which are to be explored, and the chapters following that describe the experiments performed, and the conclusions reached.

4.1.3 To build better robots

It has long been believed that introducing developmental processes into the methodology of evolutionary robotics will provide opportunities for the evolution of agents capable of more complex and more difficult behaviours (Hinton and Nowlan, 1987; Harp et al., 1989; Kitano, 1990; Ackley and Littman, 1991; Gruau, 1994). Some of the reasoning behind this belief is a little vague, and some of it is more well-founded.

It is certainly true that by copying phenomena found in nature many useful methods and systems have been found, and nature certainly makes use of many developmental systems. Thus, one of the more informal arguments for studying development is that it is likely to produce better agents because it is similar to the process that produces extremely capable agents in nature.

The following sections outline some of the arguments put forward to explain how development may be useful in the production of more capable controllers and agents.

Modularity and Symmetry

Developmental processes may allow the reuse of parts of the genotype to construct similar structures in different parts of the organism. A possible example of this kind of mechanism is that cells of the same cell type in multicellular organisms may owe their similarity to each other because they are influenced by the same regions of the genome. The physical left/right symmetry in most organisms may well represent such redundancy, and there are likely to be many more subtle symmetries involved.

Gruau (1994) exploited a developmental mechanism to produce modularity making it easier to evolve a six-legged walking robot with similar neural mechanisms controlling each leg.

Environmental influence

Often developmental processes produce an advantage by allowing environmental conditions to influence the developmental pathway.

This process may depend either on predictable or guaranteed environmental conditions (for example in some plants where the process of root growth depends implicitly on the fact that gravity pulls downwards towards the earth) or on unpredictable conditions.

Waddington (1975) discussed the case of predictable environmental conditions as a situation where influence from the environment may be used at one stage of the evolutionary process before a process of 'genetic assimilation' takes place, and development begins causing such conditions to occur, making the outcome present even if the environmental conditions which originally triggered it are removed. This leads to a Lamarckian-like process where environmental adaptations may be assimilated as genetically-specified characteristics over the course of evolutionary time.

Most forms of learning involve the process of influence from the environment, and, since the definition of development given in section 1.2 includes learning, this is an example of this kind of development. Clear similarities between the neural processes of learning and development have been shown in mice (van Praag et al., 2002).

Flexibility over different environments

In nature development is often used as a mechanism that allows flexibility in the character of the phenotype depending on factors in the environment. This is clearly a useful effect of development if the environments into which organisms are born contain variation (which is almost always the case). Non-developmental genotype-phenotype mappings are incapable of supporting this form of flexibility.

Potentially, an evolved developmental agent could be flexible enough to adapt to different circumstances, which could allow it to be used in many different contexts, and could also offer the possibility of making the transfer from an agent evolved in simulation to its implementation in hardware, since the inevitable differences between the simulation and reality could be smoothed by the flexibility of the agent.

Adaptability to changing conditions

Humans and other animals have a remarkable capacity to adapt to disruptions in their bodies and environments (Stratton, 1897, 1896; Ewert, 1930). This is probably made possible through of their ability to undergo structural change in response to changing circumstances.

There is an increasing amount of evidence to suggest that the mechanisms of lifetime adaptation in natural organisms are strongly connected with those of early development: they may even be the same mechanisms operating under different conditions. If developmental agents can be successfully evolved in an evolutionary robotics context, the developmental mechanisms operating throughout the agent's lifetime could allow the agent to adapt to the change it experiences.

These changing circumstances may be predictable (such as the changes needed to adjust for an increase in size as an organism grows, or the changing seasons), or they may be unpredictable (such as an injury to the organism). In both cases, the greater capacity of a developmental system permanently to change its properties give it the potential to deal with such situations better than a more static system.

In order to build better robots which incorporate developmental ideas, we must not only explore open questions and learn abstract knowledge about development, but we must also gain practical knowledge of how to generate developmental controllers through evolution, and we must understand how such generated controllers work in practice. The experiments described in the following chapters provide examples of how developmental controllers may be generated, and some of the lessons learned in the process are also explained. The in-depth analyses described in sections 7.2 and 9.2 may provide much useful insight into how such controllers generate different behaviours in practice.

4.2 Open Questions

The work described in previous chapters, and the motivations explained above, lead to a number of open questions about developmental systems which are addressed through the work described in this thesis. The following sections describe these questions and outline some of the reasoning behind them. Section 10.2 examines what light has been shed on them, and looks into some explanations for the results found. These experiments are exploratory only and not rigorously controlled. As such, they may not be relied upon to support any general hypotheses.

1. May developmental systems be evolved to generate simple behaviours using today's methods?

In order to be able to study evolved developmental systems performing behaviours of interest, it must be possible to generate agents capable of these behaviours. An idea underpinning this work is that systems exhibiting development may be evolved using current evolutionary techniques and within current technological limitations.

Further than that, though, it is believed that the developmental systems may ease evolution (leading to higher fitness) in some situations, and allow evolution to produce more complex behaviours than may be produced in non-developmental systems. This belief arises from the idea that development's incremental and adaptable mechanisms fit well with the process of evolution. All of the potential advantages described in section 4.1 might be utilised during evolution to produce advanced behaviours.

The primary motivation for this idea is that in the natural world development and evolution appear to be so tightly coupled as to be virtually indistinguishable, and many of the possible advantages of developmental systems such as flexibility, adaptability and use of environmental influence are seen in those systems, appearing to ease the evolutionary process and allow for more open-ended evolution.

2. Can developmental systems handle predictable changes in required behaviour better than non-developmental ones?

Natural systems are often required to alter their behaviour in predictable ways over their lifetime: for instance, most animals have to adapt their muscle control to deal with the enlargement of their limbs, and a caterpillar must alter its behaviour radically after it undergoes metamorphosis into a butterfly. It is believed that such alteration in behaviour may be accompanied by alteration in brain structure.

Similarly, in artificial agents, it may well be beneficial (in terms of maximum fitness achieved) to allow the alteration of the behaving controller in tasks requiring predictable change (change

which may be relied upon to happen in every generation) since one possible way of generating changing behaviour is to alter the structure of the behaving controller to give it different dynamics.

3. Will developmental systems selected for change in the controller at the time at which behavioural change is required evolve more successfully to exhibit that change?

The previous question includes the assumption that a change in behaviour is most easily generated through a change in controller structure. If this assumption is true, we might further expect that if selection pressure is applied to an agent to undergo change in its controller at the time at which behavioural change is required, this might encourage evolution to produce agents which undergo change in the controller at the appropriate time, leading to the production of a linked controller-behaviour change system that has higher fitness than those generated without such additional selection pressure.

4. May developmental systems be evolved to perform tasks that are too complex for any non-developmental system?

Section 4.1 outlines some of the expected advantages of using developmental controllers over more static ones. It is believed that the exploitation of modularity and symmetries, the use of environmental influence, and flexibility and adaptability will allow developmental controllers to gain higher fitness in some complex tasks than any non-developmental controller. This question is explored here by comparing a single developmental system with a single non-developmental system. Clearly there is room for much wider investigation of this question in the longer term.

5. Are developmental controllers more capable of performing learning tasks than non-developmental ones?

If a developmental controller can undergo structural change in its behaving controller, that structural change could be used to adapt the behaving controller in response to learning, which may be easier to evolve to higher fitness than a controller which adapts its responses without undergoing structural change.

Some of the pieces of work described in section 3.1 show correlation in natural systems between the performance of learning tasks and structural change occurring in the brain. This correlation leads naturally to the idea that such structural change is a useful way of bringing about behavioural change.

6. Are developmental controllers more able to produce different behaviours depending on the type of environment with which they are faced?

Given that developmental controllers may develop in entirely different ways given different environmental conditions, it is reasonable to suggest that they may be able better to produce different behaviours when faced with different environments, and thus achieve higher fitness in such scenarios.

The idea behind this question is that a developmental process may be able to use environmental triggers to produce entirely different controllers in different environments, whereas a non-developmental controller would be required to produce different behaviours in different environments with the same controller structure.

The work of Nolfi et al. (1994; 1996) showed that a developmental controller of their devising was able to adapt more effectively to differing environments than a comparable non-developmental

controller.

7. Are developmental controllers more robust to previously unencountered change than non-developmental ones?

Since the developmental process may be seen over each lifetime as a process of adaptation - moving from an unpredictable unordered state to a more ordered one, and since this process may be influenced by and take advantage of regularities in the environment, it may be that if changes occur in the body or environment of a developmental agent, the developmental process could be altered in such as way as to preserve the original behaviour of that agent more effectively than in a non-developmental agent.

The work of Waddington (1975) covers in detail the ability of natural developmental systems to maintain the stability of behavioural or morphological outcomes in the face of variation, both genetic and environmental. In the cases he covers, the fact that the systems he studies are developmental is crucial to the explanations he finds for their robustness.

8. Are developmental controllers evolved to be robust to noise in the developmental process more robust to previously unencountered change than those evolved in more predictable conditions?

Following from the previous question, the idea that the ability of an agent to self-construct from an unpredictable state to a stable state gives it stability in the face of unexpected change may be extended with the idea that if the initial unpredictability is increased, the stability of the final solution may also be increased. Thus, agents evolved to produce stable outcomes in the face of noisy developmental processes may be more stable to other types of unpredictable change than those evolved to develop through a less noisy process.

9. Will controllers whose development involves mainly guided pruning of structure be more successful than those whose development involves mainly guided growth?

Section 3.3 describes an ongoing debate between two groups in the world of neuroscience. The suggestion of selectionists that brain development in animals and humans is mainly guided by Darwinian-like selection processes leads to the idea that controllers involving pruning or death of structure, being more like those found in nature (according to selectionists), may be more successful (evolve to higher fitness) than those involving mainly growth of new structure.

This question, more than the others mentioned here, is not presented as an assumption of the author. The debate described is very much still ongoing. Furthermore, the extrapolation of the opinion of selectionists about how one existing system actually operates to the idea that in the abstract this type of system will always be best, is uncertain at best. However, if one way of working could be proved conclusively to be superior (easier to evolve to higher fitness), this would add some weight to the idea that natural evolution might have taken this path, all other things being equal.

Having enumerated the open questions which are to be explored by the experiments performed in this work, the following chapter describes the general methods used, and the following chapters describe the experiments and their results.

Chapter 5

Methods

5.1 Evolutionary robotics

The work described in this thesis is within the field of *evolutionary robotics* (Nolfi and Floreano, 2000), which is concerned with designing robots through the use of artificial evolution. Artificial evolution is a method of automatic design which is inspired by natural evolutionary processes. This section outlines some of the most important concepts in evolutionary robotics that run through the work.

Evolutionary robotics uses a bottom-up approach to designing agents capable of very simple behaviours such as navigation and simple visual object recognition. This approach was taken when it was discovered that the top-down symbol processing systems traditionally used in artificial intelligence research encountered serious problems when embodied in real robots. Evolutionary robotics takes inspiration from natural systems in the design of its robots and controllers and uses genetic algorithms (section 5.1.2) to alter the parameters automatically to satisfy general requirements specified in a fitness function (section 5.1.4). Genetic algorithms are simplified models of biological evolution.

The field has had much success in designing robots capable of simple tasks which have proved difficult for other approaches, such as navigation (Ficici et al., 1999), visual recognition of object classes (Cliff et al., 1996) and physical locomotion (Gruau, 1995) and has opened lines of enquiry of potential interest to researchers in other disciplines such as animal behaviour (Dale, 2002), neuroscience (Husbands et al., 1998) and psychology (Beer, 1996). Active progress is being made on the evolution of robots capable of adaptation to changing circumstances and disruptions to their body and environment (Di Paolo, 2000; Urzelai and Floreano, 2000). Current challenges centre around the generation of more complex behaviours such as those requiring long term retention of complex internal state, and reasoning abilities such as those which might be more naturally generated with the use of symbol processing systems.

5.1.1 Agent-based simulations

The robots being evolved in this thesis are simulated inside computer systems, and are not instantiated as real, hardware robots. This approach allows the process of evolution to progress extremely

quickly, and gives the experimenter complete control of robot and environment. This high degree of control can be something of a double-edged sword, however, since it is easy to inject one's own assumptions into a model. Careful analysis of the purpose and scope of a piece of work can reduce the risk of making assumptions that the experiment is designed to prove. A framework within which that analysis may take place, *minimally-cognitive behaviour*, is described in section 5.3.

Unlike much simulation work in the pure biological sciences and psychology, the simulations described here are *agent-based*. This means that each individual agent (simulated robot in this case) operates as an autonomous unit which acts and interacts independently. The alternative approach is to model changes in populations and proportions, whose movements may be modelled by differential equations.

The robots modelled are intended to be both *embodied* and *situated*, as described by Brooks (1991). Brooks argues that agents which are embodied, experiencing the world directly, influencing it and receiving immediate feedback, and situated, not dealing with abstractions but with the immediate reality of their sensory environment, are able to produce complex and useful behaviour without the need for high levels of internal complexity. Evolutionary robotics works on this principle, deriving simple agents that perform tasks of interest through evolutionary processes.

5.1.2 Artificial evolution and genetic algorithms

The way the agents introduced in the previous section are derived in evolutionary robotics is through *artificial evolution*. This is a process designed to capture some of the properties of natural evolution to design agents through an automatic process which can produce novel and unexpected results. The resultant agents have the advantage that the specifics of their design have not been pre-specified by the experimenter, which reduces the weight of assumptions being made when studying ways in which a certain behaviour may be produced.

The type of artificial evolution used in this work is the *genetic algorithm* (Holland, 1975; Goldberg, 1986; Mitchell, 1996), which operates on a population of individuals, each of which is generated from a set of data (called a genotype) that partially determines its properties. The initial population may consist of individuals with random genotypes, or it may be seeded with specific data. The individuals are generated using their genotype and evaluated to find their 'fitness' according to some criteria, and a new population is created containing individuals with genotypes derived from the genotypes of the original population members by (possibly) combining genotype data from more than one individual ('crossover' or 'recombination') and making slight modifications to the genotypes ('mutation'). The new population is created such that individuals with higher fitness are more likely to pass genotype data into the new population than those with lower fitness (this process is known as selection).

Each iteration of the process is called a generation, and a typical genetic algorithm goes on for many generations. Over the course of these generations, the fitness of the individuals in the population tends to increase, and often high fitness individuals eventually emerge. Using this method, researchers are able to generate individuals that satisfy certain criteria (i.e. they achieve a given fitness score) without providing an explicit design for the solution. Often the resultant designs are surprising and subtle: sometimes they may reach beyond the limits of what a human

designer could normally produce.

5.1.3 Fitness functions

The evaluation of a robot to find a fitness score is performed by applying a *fitness function* to its behaviour. This function is designed to reward robots that are good at the required behaviour. However, the design of a fitness function is more subtle that this: it must deal well with poor individuals as well as good ones, since at the beginning of the evolutionary process all individuals are likely to perform poorly.

Thus a key property of a fitness function is to be able to distinguish between very bad and quite bad solutions, and to provide a 'path' through the 'fitness landscape' (section 5.1.4) which allows evolution to move in short steps through genotype space (applying small mutations to the genotype data) on an ever-increasing path towards high fitness.

In some cases the design of an appropriate fitness function is quite difficult. For example, when a task is sequential or consists of competing priorities, it is easy to design fitness functions that create 'local optima' where the population becomes settled in an area of the genotype space that is surrounded by worse areas (most mutations lead to lower fitness), and yet is not of a particularly high fitness relative to the final required fitness. An example of this problem and a neat solution are described by Tuci et al. (2002a). In this case the solution was to bias the fitness function to reward one part of the task 'unfairly' greatly, encouraging evolution to specialise on solving that part of the task before moving on to find a full solution.

The design of fitness functions in complex situations is crucial to the success of the genetic algorithm technique, and, as will be seen later, some work is needed in this area in order to evolve developmental robots to perform non-trivial tasks.

5.1.4 Fitness landscapes and evolvability

When a fitness function is specified it generates a surface over the space of all parameters under evolutionary control, in the sense that given any set of values for the parameters, we may construct an agent with those values, and evaluate it using the fitness function. Its fitness gives the height of the surface at the point defined by the parameter values. It is often useful to think of fitness in these terms, and to see evolution as generally proceeding upward on this surface, which is known as the *fitness landscape*. Figure 5.1 illustrates this concept in the extremely simplified situation where only two parameters are under evolutionary control.

Evolvability (Wagner and Altenberg, 1996) is an important concept, but it is difficult to define in formal terms. Loosely, it refers to how easy something is to evolve - the easier it is to evolve, the more evolvable it is - but in practice this is not only highly contingent on the exact details of the evolutionary algorithm, task and fitness function in use, but also it can have very different meanings. For example, when it is said that a system is evolvable is it meant that it reaches satisfactory solutions faster, that it is able to reach satisfactory solutions in more cases than other systems (i.e. other systems fail to find any satisfactory solutions), or that the solutions found are better? If faster, does this mean in fewer generations or fewer computing cycles? If more cases, over what space of cases is this judgement being made?

Nevertheless, a degree of evolvability (in all of these senses) is necessary in an evolutionary

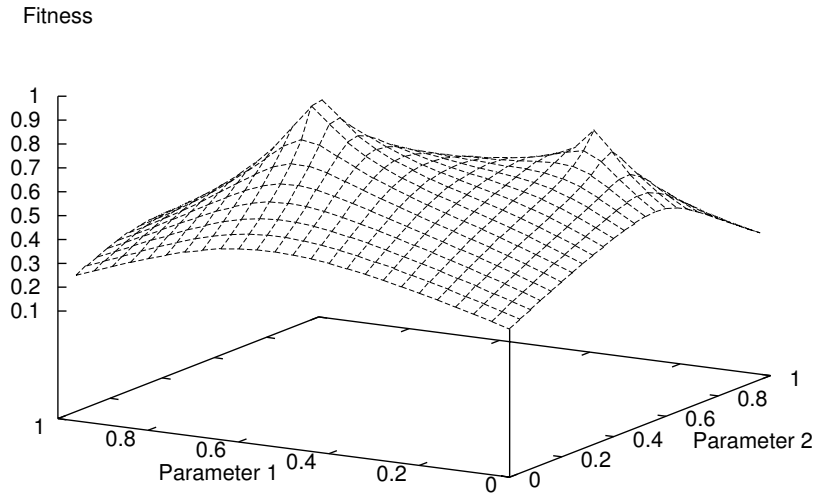

Figure 5.1: A fitness landscape in a very simple example situation. Here evolution has control over only two parameters, being able to adjust each of them between zero and one. There are two peaks in fitness, at (0.15, 0.6) and (0.6, 0.3). This graph illustrates a potential problem in the use of genetic algorithms: if evolution tends to move 'uphill' on a fitness landscape, it may settle on solutions around the lower local maximum at (0.15, 0.6) and be unable to 'escape' through a lower fitness region toward the global maximum at (0.6, 0.3). Many of the considerations in designing a fitness function are concerned with creating a landscape which does not contain many such local maxima. It is believed that landscapes over higher dimensional spaces tend to result in fewer local maxima than those in low-dimensional ones such as this.

system if any interesting or useful solutions are to be found at all. In a sense, this thesis may be viewed as the story of an attempt to design an evolvable system for producing developmental robots. Where that is the case, the goal is usually to produce agents that are better at a given task, rather than exhibiting faster or more reliable evolution.

5.1.5 Studying evolved individuals

In many cases, the products of an evolutionary process are useful in themselves (for example when evolution is used to design items as diverse as computer programs (Koza, 1991), bus routes (Chien et al., 2001) or aircraft wings (McIlhagga et al., 1996)), but, as suggested in section 5.1.1, it is also useful for generating artifacts that are the result of a relatively unbiased process but which have certain properties.

For example, it may be of interest to a cognitive scientist to examine the inner workings of a robot controller which has been evolved to cause a robot to perform a behaviour that has some cognitive significance. The goals of this thesis are along these lines.

5.2 Continuous-time recurrent neural networks

A neural network is a system that is formed of many interacting simple computational units, called neurons, each of which takes input from other neurons (or from the input to the system) and applies a non-linear function to that input before producing a single numerical output, which is passed to other neurons and/or is passed on as an output of the system. The values passed between neurons are weighted by multiplying by a value associated with a link between the pair of neurons in question.

These systems were developed as analogues of the operation of real networks of neurons in the brains of humans and animals, but they have very serious limitations as models of true brain function. Nevertheless, neural networks remain important, since, although they may be implemented in the form of computer programs, they tend to produce results of a different character from those of ordinary computer programs. For example, they may be 'trained' (by adjusting the weights based on the previous behaviour of the network) to classify normally difficult inputs such as sounds or pictures in a similar way to the way a human would classify them. They may also be used to control the behaviour of robots.

Continuous-time recurrent neural networks (CTRNNs) are a class of neural networks introduced into the field of evolutionary robotics by Beer (1996) but in use previously in the field of neuroscience. While many types of neural network are non-temporal computing devices, CTRNNs explicitly include temporal factors since each neuron acts as a leaky integrator, so that input increases its activation, or cell potential, which then gradually regresses back towards its rest value over time.

Beer showed that neural networks of this kind are capable of performing behaviours of interest in understanding cognition of which non-temporal neural networks are not capable. He explained that this is the case because a temporal network allows the robot's behaviour to depend not only on its immediate circumstances but also on the history of its interaction with its environment.

Neurons in a continuous-time recurrent neural network (CTRNN) are governed by the following equations:

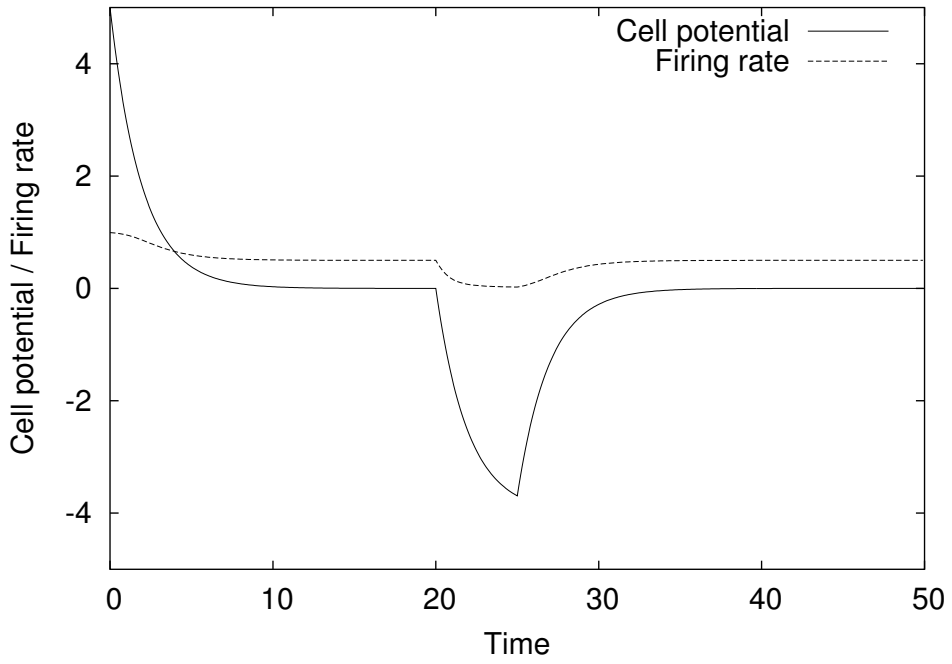

Figure 5.2: An example of the behaviour of a neuron in a CTRNN. The neuron begins with a cell potential of 5, which quickly drops towards zero. Between 20 and 25 time units it receives a constant sensory input of -4, which causes its cell potential to decrease. After 25 time units the sensory input is removed, and the neuron regresses back towards zero. The time constant of this neuron is 2. The firing rate of the neuron is shown for each cell potential value. This is constrained by its equation to be within $[0:1]$.

$$\tau_i \dot{y}_i = -y_i + \sum_j w_{ji} z_j + S_i \tag{5.1}$$

$$z_j = \frac{1}{1+e^{-(y_j+b_j)}} \tag{5.2}$$

where $\sum_j$ denotes the sum over all neurons j connected to neuron i, $y_i \in [-8:8]$ is the cell potential, $\tau_i \in [1:2]$ the time constant and $b_i \in [-5:5]$ the bias of neuron i, $w_{ji} \in [-5:5]$ is the weight of the connection from neuron j to neuron i, $S_i \in [0:1]$ is the amount of sensory input to neuron i and $z_j \in [0:1]$ is the firing rate of neuron j.

A graph giving an example of how the activation of a neuron varies over time is shown in figure 5.2.

The CTRNN controllers used in this work have a limited range of time constants (in the range $[1:2]$) that restricts them to short term dynamics. Since the developmental processes may operate over longer timescales, this produces a clear separation between the roles of the two mechanisms. The controllers used in part two are based on the CTRNN design, but since the simulation time step that is used is relatively large, they do not approximate well the behaviour of a system following these equations exactly.

Other details and structures of the controllers used in this thesis vary across different experiments. They are described in sections 6.2.2 and 8.2.2.

5.3 Minimally cognitive behaviour

The CTRNN controllers described in the previous section have been developed to deal with a class of behaviours Beer describes as *minimally cognitive*. These are behaviours which are simplified as far as possible while still containing elements of interest for those studying cognition. Beer defines these minimally cognitive systems as:

> "The simplest possible agent-environment systems that raise issues of genuine cognitive interest." (Beer, 1996)

Crucial to this approach is the application of deep analysis to evolved controllers in order to understand how they work. Beer and his colleagues work with

> "... agents whose capabilities are both rich enough to explore cognitive behaviour yet simple enough to be tractable to evolution and analysis." (Slocum et al., 2000)

They believe that by simplifying agents and behaviours to their bare essential features it may be possible to begin understanding how they work, and they see this work in the context of a long tradition of reductionist methodologies in science:

> "We believe that such simpler idealised models can serve as 'frictionless planes' in which basic theoretical principles of the dynamics of agent-environment systems can be worked out." (Slocum et al., 2000)

For example, Slocum et al. (2000) investigated the generation of 'pointing' behaviour, where a simulated robot was required to move an opaque arm to catch an object which it could sense using an array of ray sensors (designed to be similar to infrared distance sensors in real robots). The robot's controller received inputs from the ray sensor, and from the arm giving its current position, and moved the arm by providing outputs which specified the torque applied to the arm in the clockwise and anticlockwise directions. This experiment was designed to investigate the mechanisms that can bring about a correlation between a robot's behaviour and the properties of an object external to it, whilst discriminating between stimuli produced by its own behaviour and those produced by external objects. Figure 5.3 illustrates this experiment.

Of course, decisions about which behaviours are of cognitive interest are always subjective, and in the course of this thesis it may be seen that a need was identified to move onto agent-environment systems with higher complexity than some of those used by Beer and his colleagues in order to investigate the kinds of behaviour which are of interest in the study of developmental systems. There is inevitably a tension in a reductionist methodology between the drive for simplicity and the need to preserve vital factors - Beer's agents may just barely be described as embodied and situated, but more complex models have greater capacity to fulfil the meanings of those words. Beer and his colleagues also remove noise from their simulations in an attempt to make them more tractable for analysis. However, some schools of thought (Jakobi, 1998) suggest that the use of noise is crucial in evolutionary systems, not only to evolve more complex systems (which are

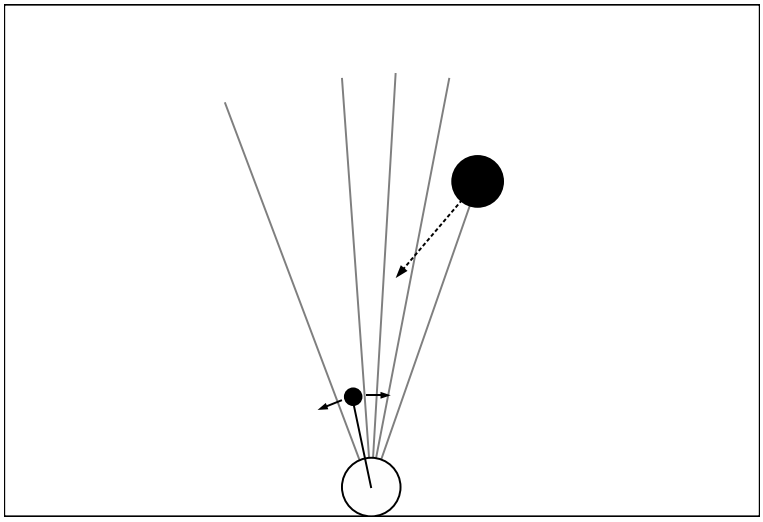

Figure 5.3: An agent performing a minimally cognitive behaviour. The agent must move its arm to 'point' at the moving object, distinguishing between sensory input generated by its own arm and by the object. Figure adapted from (Slocum et al., 2000).

perhaps of greater cognitive interest), but also in order to understand natural behavioural systems, which, of course, evolve and develop in noisy circumstances.

Two of the experiments used in part one of this thesis to investigate developmental robots are replications of two of Beer's minimally cognitive behaviours, and all of the other experiments share a similar philosophy of reducing unnecessary complexity to lay bare a behaviour of cognitive interest. However, compromises have been made between simplicity and the need for certain features to be present in order to understand different phenomena: agents capable of movement in two dimensional environments are used in order to increase the variety of available outcomes, and noise is used to aid evolution and investigate its role in evolving robust controllers. The experiments in this thesis are designed to facilitate the crucial process of deep analysis of the generated controllers in order to advance understanding of how such systems can work in practice. Such analyses are undertaken in several cases.

5.4 Statistics

Many of the results that are presented in chapters 7 and 9 involve several evolutionary runs performed under a range of different conditions. Often it is interesting to attempt to find evidence that two sets of evolutionary runs performed under different conditions result in different outcomes. In order to do this, a number (usually 10 or 20) of runs was performed under each set of conditions, and a measure of the fitness of the evolved agents was compared with the same measure for the runs performed under different conditions.

The measure that was used for most of the results discussed was an approximation of the fitness of the fittest agent present in the last generation of the evolutionary run. This was found by evaluating each agent in the last generation for 100 lifetimes, and taking its average fitness. The score of the agent with the highest score by this measure was taken to be the fitness of the entire run. In this way a single fitness score is obtained for each evolutionary run.

In order to compare two different sets of conditions each evolutionary run is taken to be a sample from a population of all possible runs under those conditions, and the two series of samples for the different conditions are compared. In general there is no reason to assume that the data are normally distributed, so the Mann-Whitney non-parametric test was used to test whether the differing conditions have an effect on the outcome. In this case the null hypothesis is that the data are drawn from identically-distributed populations. This test uses the U distribution.

The test used is two-tailed, since there is no *a priori* reason to assume that one set of conditions will necessarily produce fitter agents.

Where p values are supplied, they represent the probability under the relevant assumptions of generating the data given that the two populations have equal medians. This means that a low p value suggests that it is likely that the different sets of conditions have a material impact on the evolutionary process, making the likely fitness of evolved agents different under the different conditions. For the purposes of this thesis if $p \leqslant 0.05$ the result is considered significant, and if $p \leqslant 0.01$ it is considered highly significant.

Having outlined the methods in use throughout the thesis, the next chapter explains the motivations and specific methods used in the first piece of experimental work.

Chapter 6

Part 1 - Chemical-guided growth networks

The methods described in chapter 5 form the basis upon which this work builds. Since the goal here is to learn about development, a developmental system that may be evolved was designed. This system consists of a simple robot controlled by a controller that exhibits development. This controller is an adaptation of a CTRNN, with added features that form a separate 'layer' of dynamics which alter the number and connection of neurons, allowing it to grow from a small number of 'seed' neurons into a network that successfully controls the robot, performing several required behaviours.

Inspiration for the design of these controllers was taken from several aspects of development in natural systems, including the use of chemical gradients, spatial organisation, a single genotype for all the neurons, and dynamic alteration of structure dependent on both environmental stimuli and genetic factors.

Section 6.1 describes the reasoning behind the design of these controllers, and section 6.2 describes the detail of that design, along with the robot morphologies, environments and methods used.

6.1 Motivation

Networks of real neurons in animals develop in very complex ways, and there are a large number of phenomena one might hope to model in order to achieve some of the expected advantages of development discussed in chapter 4. This section outlines some of the phenomena that have provided inspiration for the design of chemical-guided growth networks.

Growth and change during lifetime In order for the growth process to be analogous to the growth process in real organisms, it goes on during the lifetime of the robot, and is affected by its behaviour and sensory experience. This allows the developmental process to contribute to the flexibility of the robot's behavioural repertoire by facilitating its adaptation to changing circumstances. Chemical-guided growth networks provide a mechanism by which the experience of the robot may have anything from a small effect on development through to a completely crucial effect.

Temporal and spatial factors Clearly natural behaviour is generated within the constraints of time and space. It has been convincingly argued (Beer, 1996) that introducing temporal factors

(internal state and recurrency) into controllers allows for a greater variety of cognitively interesting behaviours to be generated. Similarly, controllers featuring spatial analogues have proved useful for behaviour generation (Husbands et al., 1998). Chemical-guided growth networks model a physical space in which neurons grow and interact.

Single genotype for all neurons In biological systems the genetic material in each cell is identical. Much past work has allocated separate sections of the genotype to different parts of the controller. This is justified in terms of different genes being expressed in different cells, but this is a dramatic simplification of the real situation. The system described here allows differential expression of different parts of what is genuinely one genotype. This method offers the additional advantage that the number of neurons may be varied without the need for a corresponding change in the length of the genotype. This aspect makes chemical-guided growth networks potentially highly scalable.

Chemical gradients The mechanism by which different neurons express different parts of the genotype is based on the levels of various chemicals at its location. This feature is inspired by the use of chemical gradients in real developmental systems to push cells down different developmental trajectories. Chemical-guided growth networks simulate the presence of chemical gradients in the space in which the neurons exist. Since the chemical gradients used have properties such as neurons which are close together tending to be similar, and potential repeating structures of neurons in different areas, the controllers have potential to produce scalable and modular neural networks.

Regulation of neuron growth The controllers presented contain the concept of self-regulation of neuron growth using a system loosely inspired by Nerve Growth Factor. The growth of new neurons is suppressed after a certain number of neurons have been grown from a particular parent neuron. This provides a direct mechanism which may be used by evolution to construct developmental systems that stabilise after a certain period of time.

The system as a whole is designed to be an incremental addition to the well-known CTRNN model (section 5.2). By building the system on a model already relatively well understood in the field, the results should be easier to understand and compare with related work.

While previous work, such as that described in section 2, may have provided examples of developmental systems incorporating some of the mechanisms described above, it is believed that the combination of all these elements together and using then in an agent-based robotics scenario is a novel innovation, offering the potential for investigation into developmental phenomena not accessible to models with more limited scope.

6.2 Methods

The robots, environments and tasks described here are replications and extensions of work by Beer (1996). They are designed to allow the study of minimally cognitive behaviour (section 5.3).

In parallel with the experiments using developing robots, replications of Beer's experiments using CTRNN controllers on the tasks used were undertaken, in order to provide a means of comparison of the developmental robots with the non-developmental ones.

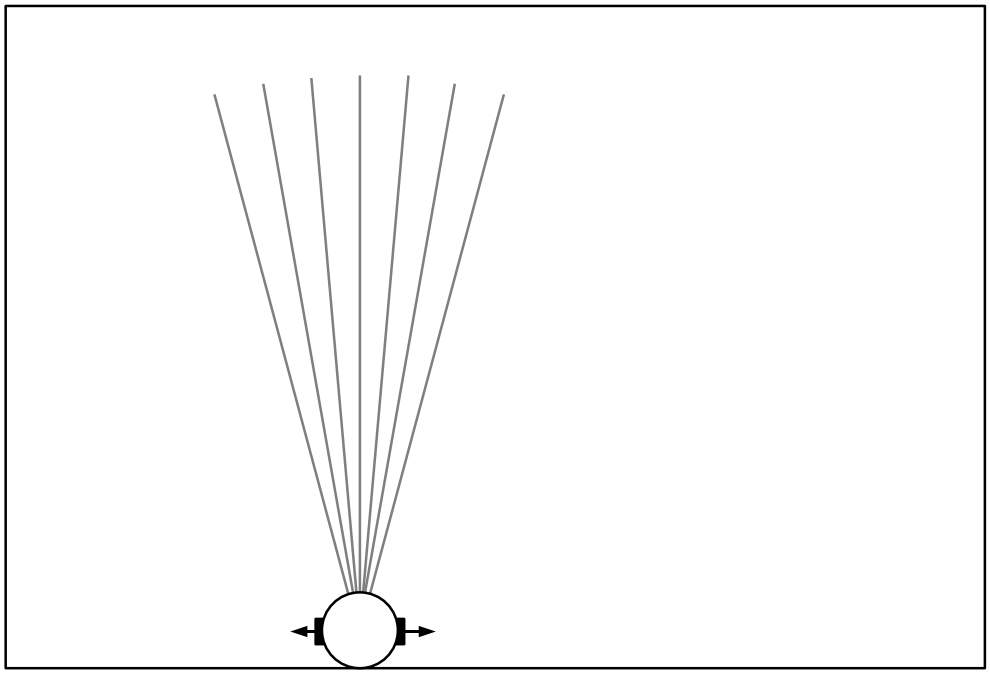

Figure 6.1: The minimal simulation robot in its environment.

6.2.1 Robot and environment

The robot, which is depicted in the diagrams as a circle, but which for the purposes of the experiments detailed here, does not have a body, has a number of distance sensors of length 220 protruding from its upper side in a fan shape covering an angular range of $\pi/6$ radians. It exists at the bottom of a rectangular environment of width 400 and height 275, and its movement is constrained to be horizontal with the vertical co-ordinate of its centre being constant at 15 units above the bottom of the environment. This setup is illustrated in figure 6.1.

The robot's horizontal velocity is determined by the following equation:

$$\dot{x} = 5(m_r - m_l) \tag{6.1}$$

where m_r and m_l are the activations of the right and left motors respectively, both $\in [0:1]$.

If the robot's centre horizontal position x becomes < 0 it is immediately reset to 0 and similarly if it becomes > 400 it is reset to 400.

The activation of each sensor is determined by the minimum distance from the centre of the robot to an object intersecting the sensor ray. Each sensor ray is 220 units long, and if the ray does not intersect any object, its activation is zero. If it does intersect an object, its activation is as follows:

$$S = 10 - \frac{d}{22} \tag{6.2}$$

where d is the distance to the closest point of intersection. Thus, $S \in [0:10]$.

In the experiments different objects fall from the top of the environment towards the bottom, and the robot interacts with them in different ways. The two shapes of object are circles of diameter

30, and 'diamonds' - squares with sides of length 30, rotated so that each side is at 45° to the horizontal. In the multiple discrimination experiments circles are of random diameter taken from a uniform distribution over the range $[25 : 35]$ and diamonds have side lengths taken from the range $[21 : 31]$.

The sensor inputs are fed into neurons in the controller (described in section 6.2.2). Uniformly distributed noise of 10% is applied to the sensor and motor outputs at each time step.

6.2.2 Controller

In all cases the robot's controller is based on a continuous-time recurrent neural network which uses neurons with internal state, their activity decaying over time. Experiments were performed with ordinary CTRNN controllers, and with developmental controllers which add new features onto the CTRNN model. The equations governing the behaviour of neurons are given in equations 5.1 and 5.2. For the orientation task, both dynamic and non-dynamic CTRNN controllers were used, where dynamic means the time constant of each neuron is allowed to be within the range $[1 : 2]$ and non-dynamic means the time constants were all fixed at a value of 1. In the discrimination task only dynamic CTRNNs were used. There is no noise added to neuron cell potentials or firing rates.

The constraint of neuron time constants to a small range is designed to allow the neuron activity to be viewed as the behaving controller, constrained to operate over relatively small timescales, while the actions of neuron and synapse growth make changes over longer timescales, meaning they are more likely to be naturally described as part of the developing controller.

The y values from those equations are initialised to 0 and the integration is approximated using the forward Euler method with a step size of 0.1. Each neuron is either bound to one sensor, or no sensors.

The developmental networks are formed by allowing a CTRNN network to grow during the lifetime of a robot, dependent on its experience of its environment. In order to allow this to happen the CTRNN model has been expanded in several ways.

In this expansion process, an attempt has been make to strike a balance between biological inspiration and pragmatic assumptions about what kind of controllers will evolve to generate useful behaviours in a reasonable time.

In some of the experiments described below, the development process is not constrained to be symmetrical and thus produces asymmetric controllers. This is an important difference from the controllers used in Beer's (1996) work. In other experiments the developmental system is constrained to be symmetrical, but variations entering the system through noise and sensory experience mean that asymmetrical controllers occur. In every case the standard CTRNN controllers are constrained to be symmetrical.

The chemical guided growth network consists of a two-dimensional space of size 1×1 (the units are arbitrary). Certain areas of the space correspond to the robot's sensors (neurons within those areas receive input from the corresponding sensor) and certain areas correspond to motors (the sum of the firing rates of neurons in these areas produces the activation of the corresponding motor) as illustrated in figure 6.2. The controller is seeded at the beginning of the robot's life with one neuron in each sensor region. Neurons have radius 0.05 or 0.1.

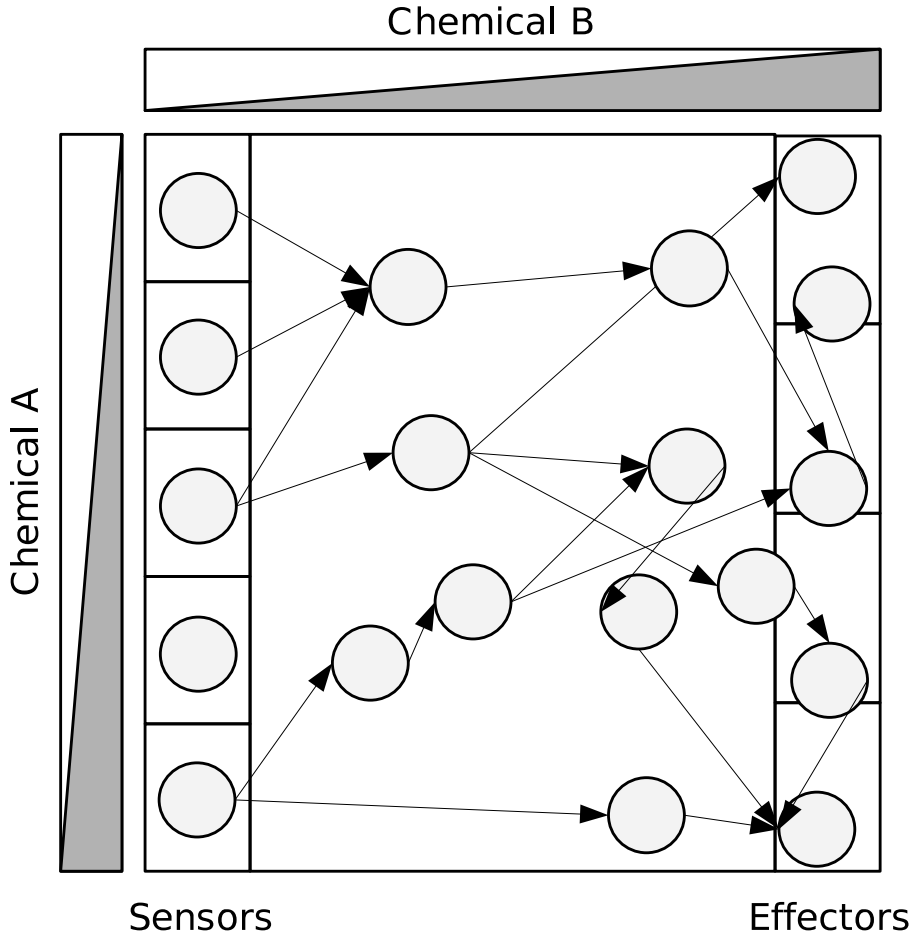

Figure 6.2: The controller is a two-dimensional space with two chemical gradients over it, and specific sensor and motor regions.

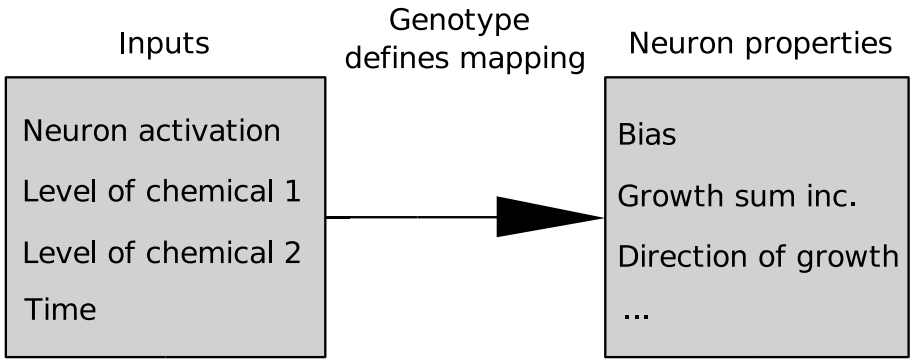

Figure 6.3: The robot's genotype defines a mapping for each neuron from its activation, the chemical levels at its location, and the current time, to its properties.

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

The growth of a new neuron is determined by two factors. First the originating neuron must have enough energy to be able to grow a child, and second its internal measure of whether to produce offspring (its 'growth sum') must be above a threshold.

The growth sum is zero when a neuron is born, and is altered according to the value specified at each time step by the relevant section of the genotype acting on the current chemical levels, the time, and the neuron's cell potential.

If the neuron has enough energy and its growth sum goes over the threshold, another neuron is grown and the parent neuron's energy is reduced. The new neuron's distance from its parent and the angle at which it grows are found by applying the relevant part of the genotype to the chemical levels and cell potential of the parent neuron and the time.

The parent neuron is linked by a synapse to the child neuron. The strength of this synapse is determined by another section of the genotype acting on the conditions of the parent neuron.

If a neuron is about to grow in a position where it will overlap with another neuron (in the two-dimensional space they inhabit) then a new neuron is not grown and the would-be parent simply grows a new synapse to the neuron already at that position, if there is not already such a synapse. All neurons are the same, fixed size, although this size was varied from experiment to experiment.

The genotype mapping functions are designed to approximate arbitrary functions. There are 7 properties determined in this way for each neuron: bias, time constant, energy, growth increment, direction of growth, distance of growth and synapse weight. The genotype is shared by all neurons in the controller and its size is the same for any number of neurons. The relationship between the genotype, its inputs, and the properties of neurons is illustrated in figure 6.3.

The time input is calculated as follows:

$$x_t = 0.5 + 0.5\sin(0.1t) \tag{6.3}$$

where t is the time elapsed since the birth of the robot.

The functions used as genotype mapping functions in the experiments take a number of forms, all designed to be easily manipulable into whatever shape the evolutionary process requires to achieve the desired developmental trajectory.

The first form of genotype mapping function is a sum of sines:

$$p = \sum_{i=1}^{N_c} \sum_{j=1}^{N_s} \left(a_{i,j,1} \sin \left(a_{i,j,2} x_i + a_{i,j,3} \right) \right) \tag{6.4}$$

where p is a property of the neuron, the $a_{i,j,k}$ are genetically-determined values, x_i is the local concentration of chemical i or the time input or the cell potential, N_c is the number of inputs (2 chemicals + 1 for time + 1 for cell potential) and N_s is a constant value of 10. The use of a function of this form is inspired by the Fourier Transform: for sufficiently large values of N_s a function of this form may approximate any continuous function.

In this case the genotype is a list of 840 real values, since $7 \times N_c \times N_s \times 3 = 840$. The 7 occurs because there are 7 properties: bias, time constant, energy, growth increment, direction of growth, distance of growth and synapse weight.

The second form of the genotype mapping function is a polynomial of order 3:

$$p = \sum_{i,j,k=1}^{N_c} a_{i,j,k} x_i x_j x_k + \sum_{i,j=1}^{N_c} b_{i,j} x_i x_j + \sum_{i=1}^{N_c} c_i x_i \tag{6.5}$$

where p is a property of the neuron, $a_{i,j,k}$, $b_{i,j}$ and c_i are genetically-determined values, x_i is the local concentration of chemical i or the time or the cell potential and N_c is the number of inputs (2 chemicals + 1 for time + 1 for cell potential).

The third form of the genotype mapping function involves interpolating a grid of explicitly-provided values evenly distributed within the input space. The value at each grid point is provided by the genotype, and the interpolation is linear. Further description of the benefits of this form of function and the details of its use may be found in section 7.1.3.

For all of the above functions, the genotype may be thought of as defining a vector function, taking 4 inputs (chemical 1, chemical 2, time and cell potential) and having 7 outputs. In some of the experiments below, the time input was not included.

These functions, the first taking a similar form to a Fourier transform, the second being a general polynomial, and the third being a direct interpolation, provide different types of flexibility for the genotype to determine the shape of the vector functions it defines. They are all designed to provide sufficient flexibility, and smooth enough transitions between function shapes over parameter changes, to allow evolution to search the space of possible vector functions to find those which generate the required behaviour in the agent. The use of different functions reflects the fact that very little is known about what types of flexibility might be useful, and so a variety was chosen to allow the most appropriate one to be used in each situation.

The time input is present to allow the developmental process to guarantee that at least one input will vary over time, even, for example, in a neuron linked to a sensor which receives no sensory input, and has no incoming synapses. With no time input, such a neuron may only grow a

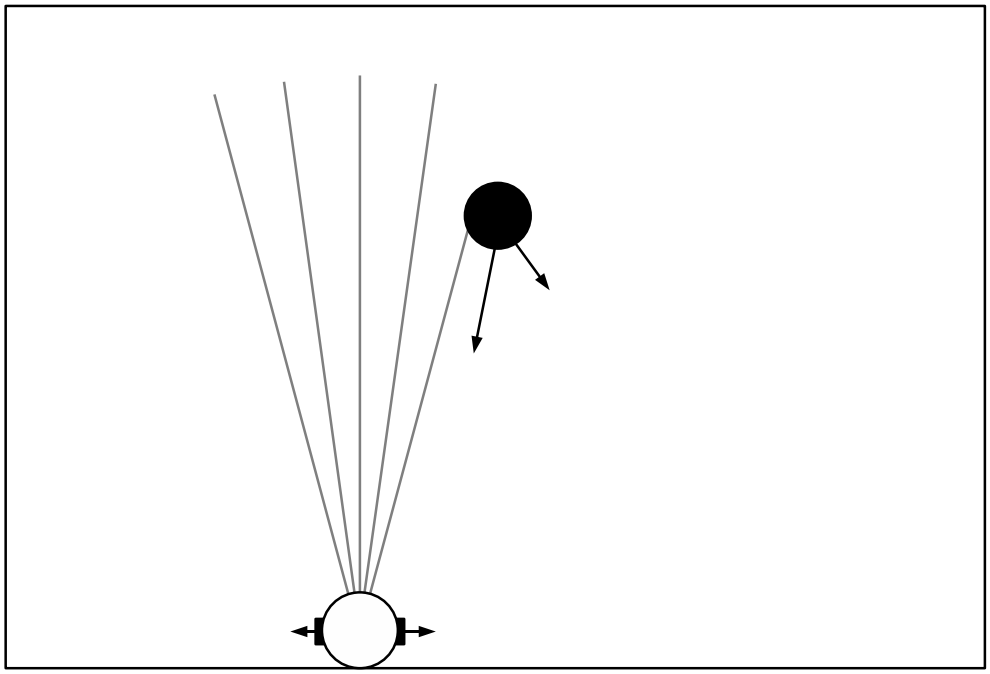

Figure 6.4: The orientation experiment. An object falls at one of various different speeds and angles toward the robot, which must orientate itself to that object.

single child neuron, since the angle and distance at which such a child would grow would be fixed. When a time input is introduced, a neuron may grow several child neurons at different locations even with no change in sensory or synaptic input.

6.2.3 Genetic algorithm

A generational, asexual genetic algorithm using rank selection with elitism on a population size of 30 or 50, with real-valued genotypes was used. In CTRNNs mutation was performed by adding a random displacement vector whose direction was uniformly distributed on the M-dimensional hypersphere and whose magnitude was a Gaussian random variable with mean 0 and variance 1 (Beer, 1996). In developmental networks mutation involved using one or both of two mutation methods: adding a random displacement vector as above but with variance 0.05 and randomising selected values on the genotype.

6.2.4 Orientation task

The orientation task tests the ability of robots to orientate themselves to circular objects falling at different angles and speeds from the top of the environment towards the bottom. This task is of interest since orientation is required as a prerequisite for many other behaviours.

Fitness for a single presentation is awarded as follows:

$$F_p = 1 - \frac{d}{W} \tag{6.6}$$

where d is the distance between the robot and the circle at the end of the presentation and W is the width of the environment (i.e. the maximum possible distance).

The robot has 5 distance sensors, and circles fall at velocities of (-5, 5), (-2, 2.5), (-1, 1.66), (-2, 5), (-0.2, 1), (1, 5), (1, 2.5), (3, 5), (4, 5), (1, 1) per simulated second, with the robot being evaluated in all 10 situations for every trial, with its controller reset before each circle falls.

In the CTRNN trials the robots have controllers consisting of five input neurons fully connected to two motor neurons.

The final fitness of a robot is found by combining fitnesses for each presentation as follows:

$$F = 0.45f_m + 0.45f_w + 0.1f_c \tag{6.7}$$

where f_m is the mean fitness over the presentations and f_w is the fitness achieved in the worst (lowest fitness) presentation.

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

There is no noise on the sensors, motors or neurons, and neurons are of size 0.05.

The sum of sines genotype mapping function was used for this task, and the time input was not present, reducing the length of the genotype to 630 values. A population size of 50 was selected, with elitist fraction of 20%, and mutation of the developmental networks involved the randomisation operation only, randomising 5 values from the genotype of 630 within their allowed ranges.

This experiment is designed to shed light on open question 1 of section 4.2.

6.2.5 Discrimination task

The discrimination task requires robots to distinguish between different falling objects on the basis of their shape. This behaviour is a simple form of classification behaviour. The robots have 7 distance sensors.

In the CTRNN trials there are 7 input neurons fully connected to 5 inter-neurons which are themselves fully interconnected, and fully connected to the 2 output neurons.

There is no noise on the sensors, motors or neurons, and neurons are of size 0.05.

Each trial consists of 20 single-presentation lifetimes between each of which the robot's controller and position are reset. Objects are dropped from 10 positions equally spaced between the point 50 units to the left of the robot and 40 units to the right, at a speed of 3 units per simulated second. At each position a circle and a diamond are dropped in separate presentations. A small random offset ($\in [-1:1]$) is added to the horizontal position of the object at the beginning of each presentation.

(Note that these conditions are slightly different from those used by Beer, where circles are dropped at slightly different positions from diamonds. This change was made to the setup to overcome problems with the different positions being used to distinguish objects, rather than their shape.)

The fitness of a robot in a single presentation is awarded as follows:

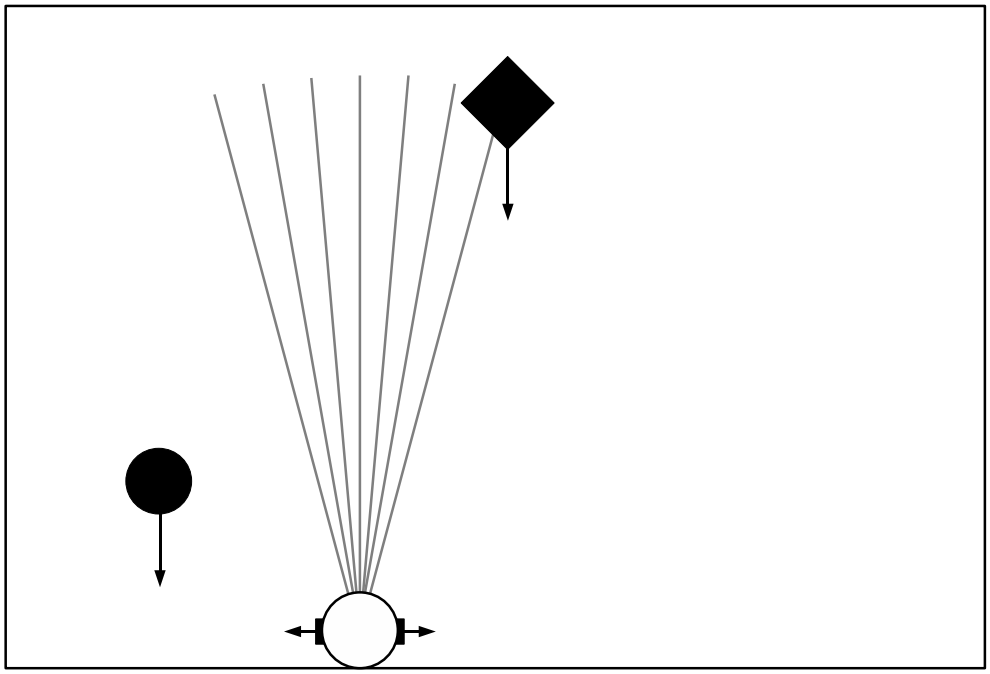

Figure 6.5: The discrimination experiment. An object (either a circle or a diamond) falls vertically at one of several possible positions, and the robot must catch circles and avoid diamonds. Note that circles and diamonds fall separately, and are shown together here merely to illustrate the fact that either may fall.

$$F_p = \begin{cases} 1 - \frac{d}{D} & \text{when the object is a circle} \\ \frac{d}{D} & \text{when the object is a diamond} \end{cases} \quad (6.8)$$

where d is the distance between the robot and the object at the end of the presentation (clipped to be $\leq$ D) and D is a maximum distance of 200 units.

The fitness of a robot over several presentations is calculated with a bias towards the worst presentation, and with additional fitness coming from the controller (as in the orientation task):

$$F = 0.8 f_m + 0.1 f_w + 0.1 f_c \quad (6.9)$$

where f_m is the mean fitness over the presentations, f_w is the fitness achieved in the worst (lowest fitness) presentation and f_c is the controller fitness as defined in the previous section.

The sum of sines genotype mapping function was used for this task, and the time input was not present. A population size of 50 was selected, with an elitist fraction of 20%, and mutation of the developmental networks involved the randomisation operation only, randomising 5 values from the genotype of 630 within their allowed ranges.

This experiment is designed to shed light on open question 1 of section 4.2.

6.2.6 Multiple discrimination task

Robots with chemical-guided growth networks evolved for the discrimination task as described in the previous section were unable to perform the task more than once in their lifetime. In order

to evolve developmental robots capable of discriminating many objects in a lifetime, the multiple discrimination task was developed. This task presents the robot with several objects sequentially during one lifetime, and fitness is awarded for correctly discriminating all of them. This was evolved for developmental networks only, since CTRNNs evolved for the discrimination task were able to perform this task trivially (see section 7.1.3).

There is 10% noise on both sensors and motors, and neurons are of size 0.1. Additionally, the size of the falling objects is varied, as described in section 6.2.1.

A fitness evaluation consists of 8 lifetimes, each of which consists of 8 falling objects, the first 2 of which are ignored.

To calculate fitness, a measure of correlation is used. Three possible conditions are identified. First, if a robot's centre is within 10 units of the centre of the object at the end of a presentation, it is deemed to have caught the object. Second, if its centre is more than 20 units away from that of the object, it is deemed to have avoided it. Otherwise it is deemed to have done neither.

Over a lifetime of 6 evaluated presentations, the robot accumulates several scores: the number of circles it caught, the number of diamonds it caught, the number of circles it avoided, the number of diamonds it avoided, and the number of times it did neither. From these it is possible to calculate the following values: N_{corr}: the number of times it caught a circle, C_{either}: the number of times it caught either a circle or a diamond, C: the number of circles that were dropped, $N_{something}$, the number of times it either caught or avoided, whether or not that action was correct, and N, the number of evaluated presentations in the lifetime.

Its fitness is calculated as follows:

$$f = \beta\left(\frac{NC_{corr} - C_{either}C}{C(N_{something} - C)}\right) \tag{6.10}$$

where

$$\beta = \frac{N_{something}}{N} \tag{6.11}$$

is a penalty term to prevent neither catching nor avoiding being a good strategy, and the rest of the formula measures the correlation between the robot's behaviour and the required behaviour.

This value is clipped to be ≥ 0.

The grid genotype mapping function was used for this task, and the time input was included. A population size of 30 was selected, with 10% elitist fraction, and mutation of the developmental networks involved both the randomisation and displacement vector operations. Randomisation involved taking each value on the genotype independently, and randomising it with probability 0.3%.

This task was performed trivially by robots with CTRNN controllers evolved for the ordinary discrimination task (see section 7.1.3) and only robots with chemical-guided growth controllers were evolved specifically for this task.

This experiment is designed to shed light on open questions 1 and 4 of section 4.2.

The following chapter details the results of the above experiments and provides analysis and discussion of these results.

Chapter 7

Part 1 - Results, Analysis and Discussion

7.1 Results

7.1.1 Orientation task

For the orientation task 20 evolutionary runs of 2000 generations each were performed with each of the two CTRNN controller types, and another 20 runs were performed with the developmental controllers. The best dynamic CTRNN run produced an individual that scored 82% of the maximum possible fitness. 8 of the 20 developmental controller runs scored more highly than this, and the best developmental run scored a fitness of 98%. The non-dynamic CTRNN controllers scored much lower fitness.

Individual robots from the ends of both the dynamic CTRNN runs and the developmental runs, when observed, clearly perform the orientation behaviour. The developmental controllers are able to react faster since their controller structure is more specialised, and they achieve greater accuracy. The non-dynamic controllers often fail to orientate themselves to the object since when it moves out of sensor range they stop moving.

The fitness graphs of the best CTRNN run and the best developmental run are shown in figure 7.1. The mean fitnesses of the three controller types over 20 evolutionary runs are shown in figure 7.2, and detailed fitnesses are shown in table 7.1 and illustrated in figure 7.3. Histograms of the scores are shown in figure 7.4.

The Mann-Whitney test shows that there is significant evidence to suggest that the chemical-guided growth networks perform better than dynamic CTRNNs, which in turn perform better than the non-dynamic CTRNNs (chemical-guided growth vs. dynamic CTRNN $p = 0.0018$, chemical-guided growth vs. non-dynamic CTRNN $p = 1.5 \times 10^{-11}$, dynamic vs. non-dynamic CTRNN $p = 1.5 \times 10^{-11}$).

7.1.2 Discrimination task

In the discrimination experiments 10 evolutionary runs of 2000 generations each were performed with the CTRNN controllers, and another 10 runs were performed with the developmental controllers.

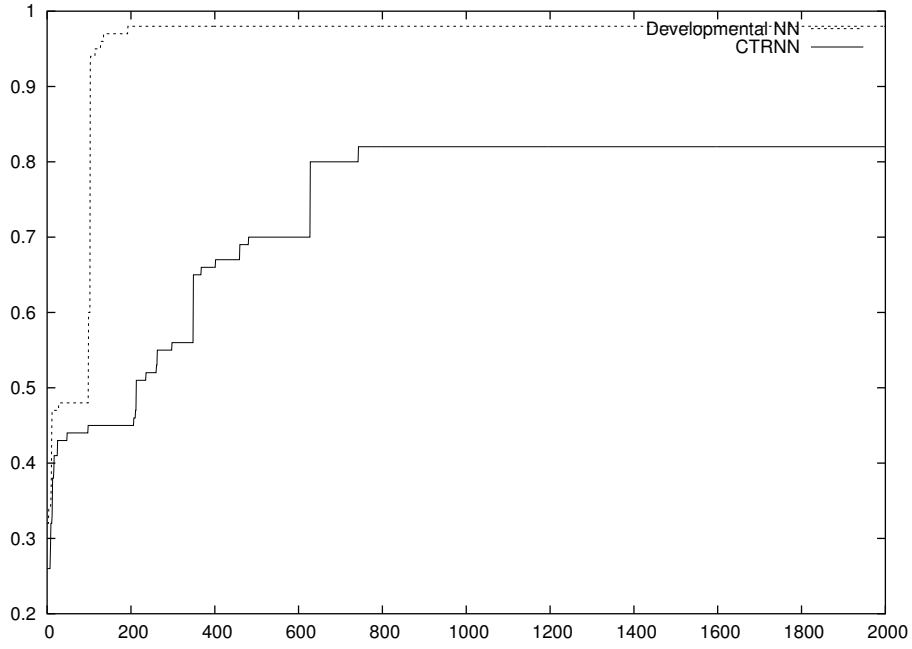

Figure 7.1: Fitness vs. generation. The best fitness achieved with a developmental controller in the orientation experiment far out-performed the best achieved by a CTRNN.

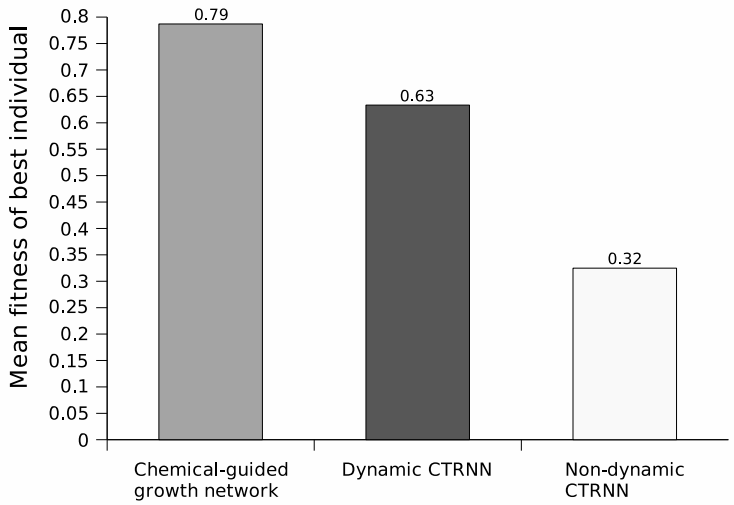

Figure 7.2: Orientation task: Each robot from the population at generation 2000 of each of the 20 evolutionary runs performed for chemical-guided growth networks, dynamic CTRNNs and non-dynamic CTRNNs was evaluated over 100 lifetimes. The mean fitness of the best performing robot (over the 100 lifetimes) from each population was taken as the score for that evolutionary run, and the bars in the figure above show the mean score over 20 evolutionary runs for each type of controller.

Controller	**Mean**	**Fittest robot**				
Chemical-guided growth network	0.79	0.92	0.92	0.98	0.87	0.74
		0.94	0.83	0.90	0.87	0.59
		0.73	0.94	0.48	0.80	0.60
		0.55	0.64	0.82	0.95	0.67
Dynamic CTRNN	0.63	0.50	0.45	0.82	0.46	0.78
		0.54	0.48	0.46	0.75	0.80
		0.78	0.79	0.79	0.50	0.78
		0.48	0.48	0.79	0.45	0.80
Non-dynamic CTRNN	0.32	0.32	0.32	0.33	0.33	0.33
		0.33	0.33	0.32	0.32	0.33
		0.32	0.32	0.33	0.32	0.32
		0.32	0.32	0.32	0.32	0.33

Table 7.1: Orientation task: The highest fitnesses achieved using different controller types. Each experiment was run 20 times, and the mean fitness over 100 lifetimes of the best individual from generation 2000 of each of the runs is shown in the right column, with the mean of these values shown to their left.

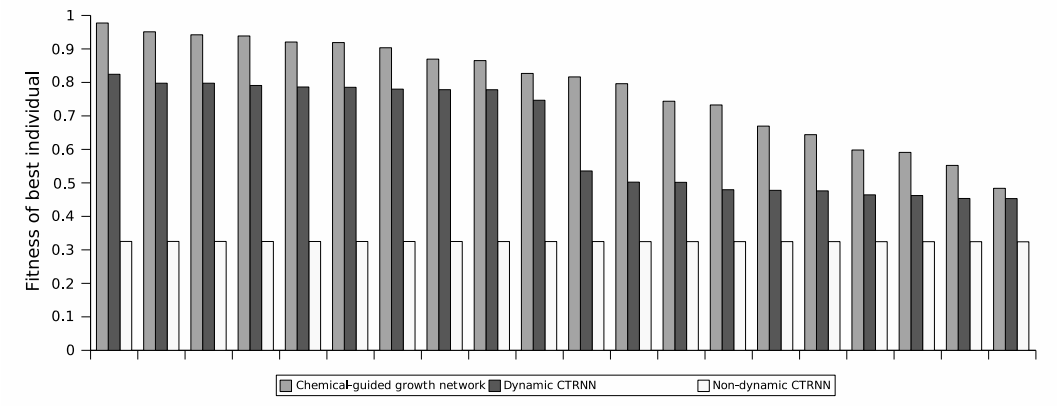

Figure 7.3: Orientation task: Fitnesses of the fittest robot from generation 2000 of each of the 20 evolutionary runs performed with each controller type. The scores are sorted in descending order to aid comparison. Exact values are shown in table 7.1.

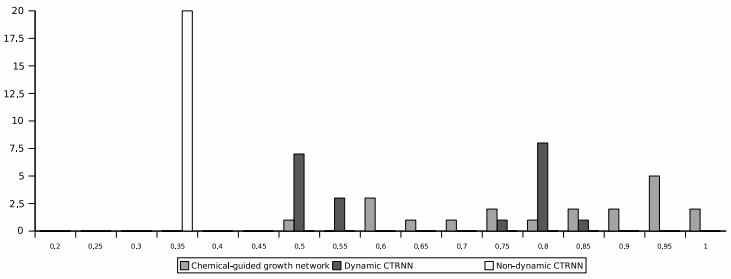

Figure 7.4: Orientation task: Histograms of the fitness scores shown in table 7.1. Where a column is labelled e.g. 0.6, it shows how many evolutionary runs resulted in robots whose mean fitness was $\in (0.55 : 0.6]$.

Controller	**Mean**	**Fittest robot**				
Chemical-guided growth network	0.75	0.71	0.71	0.71	0.72	0.73
		0.76	0.76	0.78	0.79	0.79
Dynamic CTRNN	0.86	0.77	0.79	0.80	0.82	0.85
		0.85	0.86	0.92	0.96	1.00

Table 7.2: Discrimination task: The highest fitnesses achieved using different controller types. Each experiment was run 10 times, and the mean fitness over 100 lifetimes of the best individual from generation 2000 of each of the runs is shown in the right column, with the mean of these values shown to their left.

The developmental controllers performed reasonably well, with the best robots, when examined manually, clearly correctly discriminating in approximately 75% of the presentations. However, CTRNN controllers performed better at this task, with the best controllers correctly discriminating in approximately 90% of the presentations.

The mean fitnesses of the three controller types over 20 evolutionary runs are shown in figure 7.5, and detailed fitnesses are shown in table 7.2 and illustrated in figure 7.6. Histograms of the scores are shown in figure 7.7.

The Mann-Whitney test shows that there is strong evidence to suggest that the chemical-guided growth networks perform worse than the dynamic CTRNNs ($p = 1.3 \times 10^{-4}$).

7.1.3 Multiple discrimination task

As may be seen from figure 7.8, CTRNN controllers evolved for the ordinary discrimination task were able to perform the multiple discrimination task without modification. When it was evolved, each lifetime consisted of a single presentation, but here the robot behaved for 8 lifetimes each consisting of 8 presentations (either a falling diamond or a circle). It is clear from this figure that the robot moves towards the centre at the end of most of the circle presentations, and away from the centre at the end of all diamond ones, showing that it performs discrimination behaviour.

Since this task appears to be trivial for CTRNN controllers evolved for the previous task, evolution was only performed for chemical-guided growth controllers.

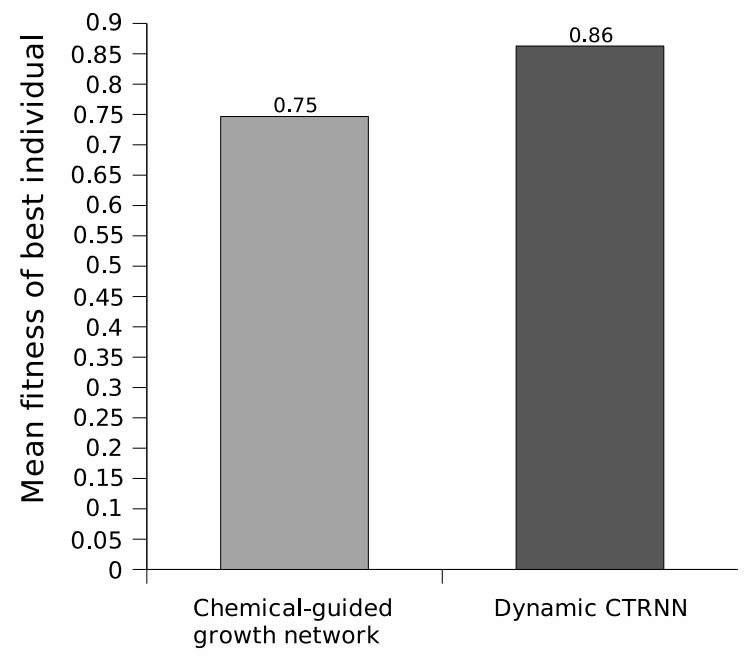

Figure 7.5: Discrimination task: Each robot from the population at generation 2000 of each of the 10 evolutionary runs performed for chemical-guided growth networks and dynamic CTRNNs was evaluated over 100 lifetimes. The mean fitness of the best performing robot (over the 100 lifetimes) from each population was taken as the score for that evolutionary run, and the bars in the figure above show the mean score over 10 evolutionary runs for each type of controller.

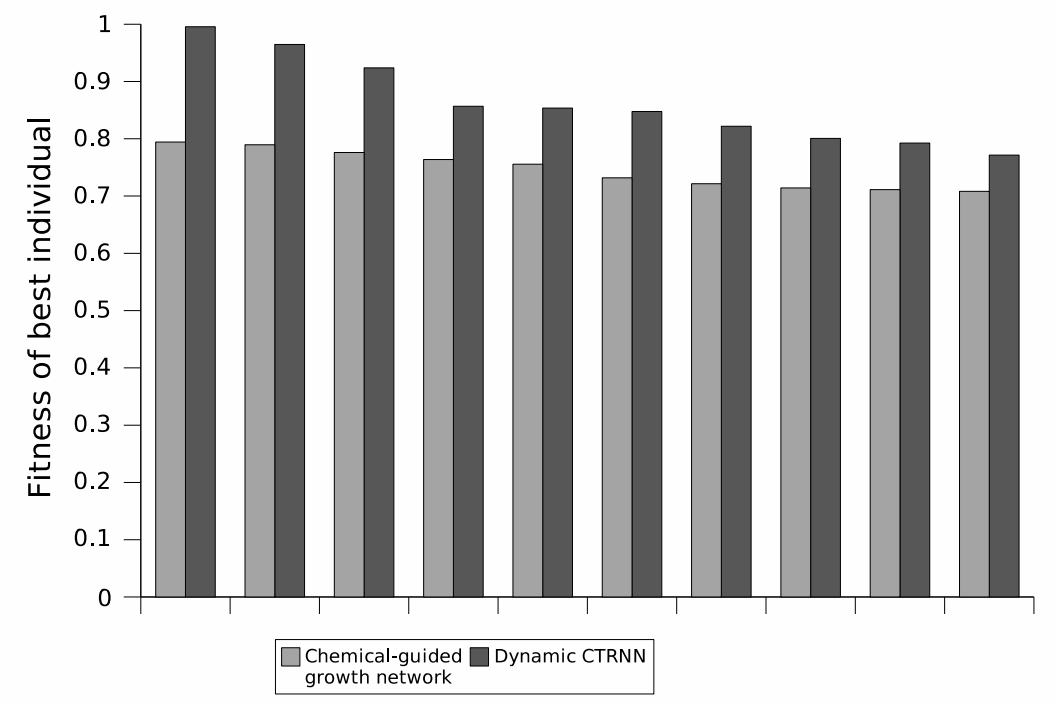

Figure 7.6: Discrimination task: Fitnesses of the fittest robot from generation 2000 of each of the 10 evolutionary runs performed with each controller type. The scores are sorted in descending order to aid comparison. Exact values are shown in table 7.2.

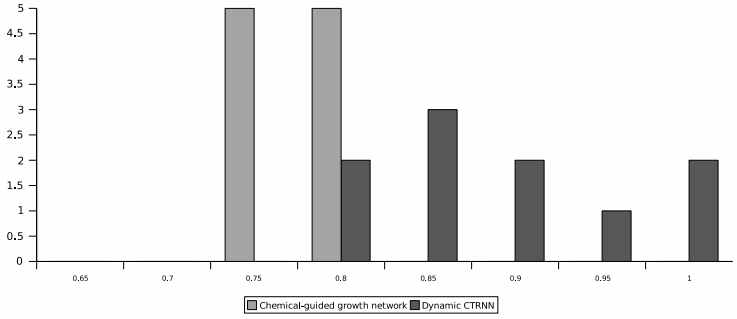

Figure 7.7: Discrimination task: Histograms of the fitness scores shown in table 7.2. Where a column is labelled e.g. 0.6, it shows how many evolutionary runs resulted in robots whose mean fitness was $\in (0.55 : 0.6]$.

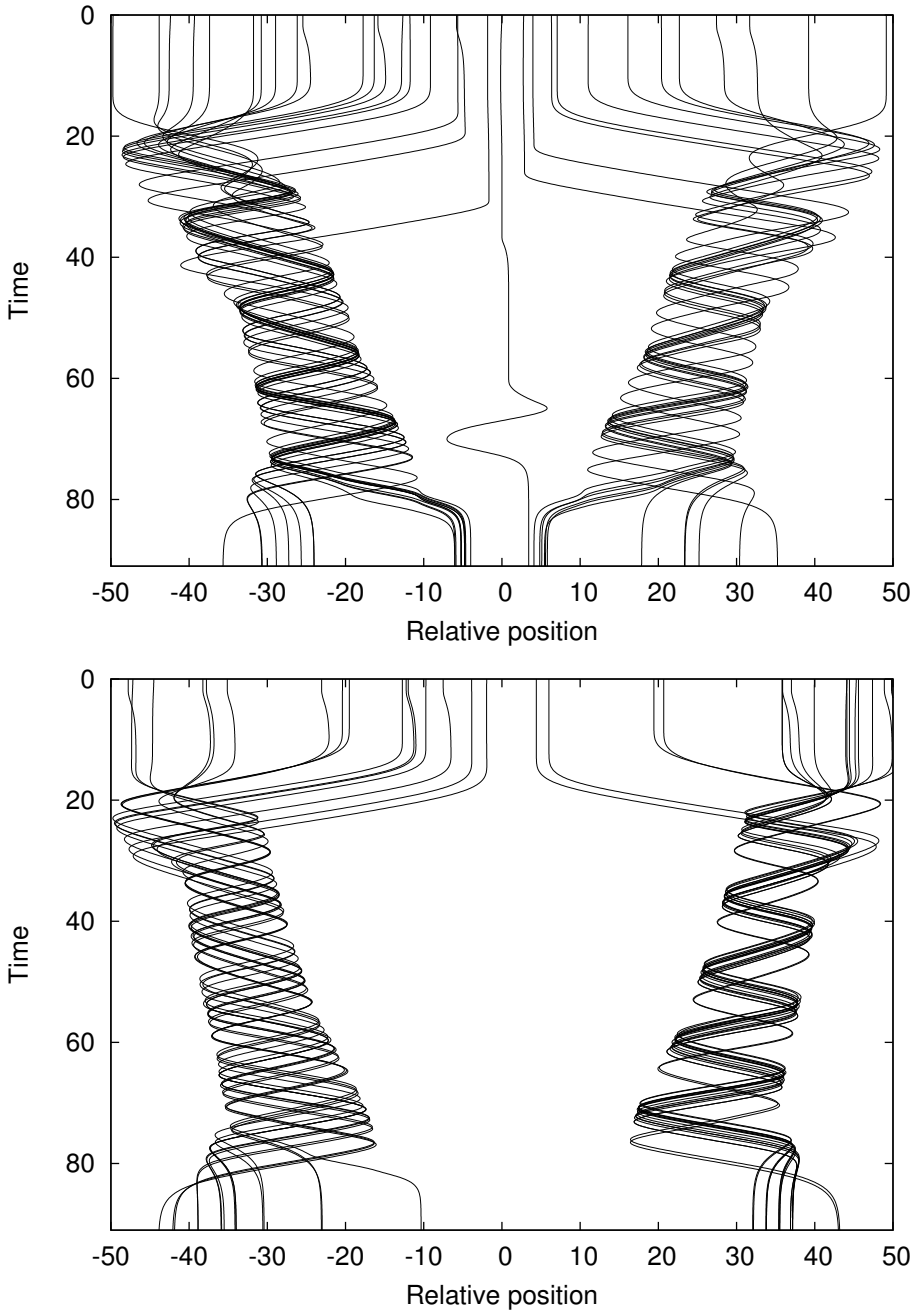

Figure 7.8: The behaviour of a robot with a CTRNN controller evolved to perform the ordinary discrimination task but here behaving in the multiple discrimination environment. Each figure shows the difference between the horizontal position of the robot and the falling object over time. The behaviour of the robot in presentations containing a circle are shown in the top figure, and its behaviour in those containing a diamond are shown in the bottom figure.

The multiple discrimination task proved to be much more difficult to evolve for chemical-guided growth controllers than originally expected. Some possible reasons for this are discussed in section 7.3. Eventually, the behaviour was evolved, but a significant amount of adjustment and 'tweaking' was required.

First, an alternative genotype mapping function (see section 6.2.2) was used. A process of trial and error resulted in the discovery that directly encoding a $4 \times 4 \times 4 \times 4$ grid of values and interpolating between them appeared to produce the best results. Thus the function that was used is as follows:

$$p = \sum_i A_i (1 - d_i/D) \tag{7.1}$$

where the A_i are genetically set values at the corners of the grid hypercube containing the point x whose co-ordinates are the 4 inputs (chemicals 1 and 2, time and cell potential), d_i is the Euclidean distance from x to the ith corner, and $D = 2$ is the distance from one corner to the opposite corner of the hypercube.

The use of this more direct encoding seemed to make the controllers more evolvable in relatively small tasks like this, but is impractical for more complex tasks due to the fact that the number of genetically encoded numbers increases very rapidly with the granularity of outputs required.

Second, the controllers were constrained to be symmetrical by altering the chemical gradient of chemical 2 so that it was symmetrical in the line dividing the controller space horizontally across the middle and reversing the angle of growth of any neuron growing from a neuron below that line. This meant that given symmetrical input the controller would grow symmetrically. With asymmetrical input, however, asymmetrical structures were allowed (and likely to occur).

Adding symmetry to the development process made it much easier for evolution to generate symmetrical structures in the controller. Since the task being used required entirely symmetrical behaviour, this encouragement to generate appropriate structures allowed evolution to find good solutions more quickly. It also prevented evolution from becoming trapped in a local maximum of simply moving in one direction or the other without reference to sensor input. This behaviour had previously been a common outcome.

Third, two presentations were added at the beginning of the lifetime which were not evaluated. This gave the robot time to develop freely without evolutionary pressure to settle quickly on a (potentially suboptimal) solution.

Fourth, incremental evolution was used. The robot was initially evolved in a situation where the objects were dropped directly above its starting position, with a uniform random offset of ± 1. This was continued for 5,000 generations, and then for the next 10,000 generations the random offset was gradually increased to ± 50. Note also that relatively very large numbers of generations were required to evolve robots able to perform this task.

Finally, and probably most importantly, a highly specialised fitness function was developed that rewarded robots for correct behaviour, but strictly did not reward incorrect behaviour that could lead to a local maximum. This was achieved by measuring the mathematical correlation between the robot's behaviour and its required behaviour.

By using this fitness function, many of the local maxima which were previously being found by evolution were avoided. However, this meant that the evaluation of robots was extremely harsh:

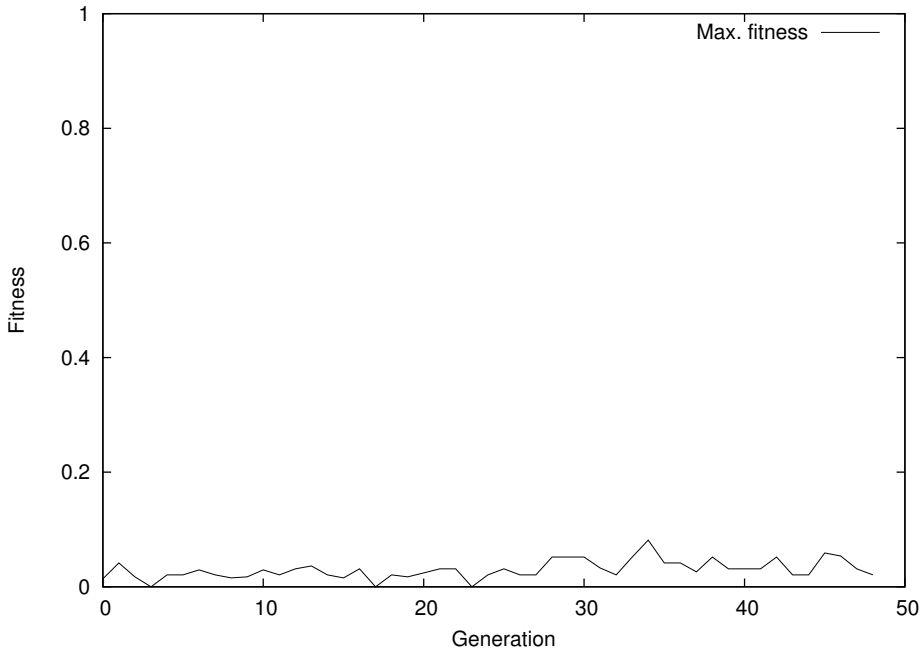

Figure 7.9: The maximum fitness achieved in each generation during the first 50 generations of an evolutionary run of developmental neural network controllers evolving to perform the multiple discrimination task.

populations could often evolve with a fitness of almost zero for 50 or more generations, essentially using random search to find a strategy that satisfied such stringent requirements. This situation is demonstrated in figure 7.9.

By using all of these strategies, robots which can perform the task well (achieving a fitness score greater than 0.5) are evolved in approximately 40% of evolutionary runs. The fitness graph of the best run is shown in figure 7.10. Interestingly, the evolved solutions seem to use very small neural networks with very fast development near the beginning of the robot's lifetime. These networks are largely unaffected by the robot's experience. This is not unexpected since the task does not require sensitivity to the conditions in the environment.

Whilst these results were hard to come by, it seems that by allowing evolution to choose the number of neurons and connections in the controller, novel, efficient solutions to the discrimination problem have been found. For example, the best evolutionary run produced robots with controllers consisting of only 9 neurons (7 inputs and 2 outputs). These controllers appear to use subtle features of the dynamics of rapid scanning over the point of the diamond as opposed to the edge of the circle to flip itself into one of two dynamic attractors leading to either catching or avoiding behaviour. One of these controllers is illustrated in figure 7.11. In contrast, the CTRNN controllers usually used to solve this problem have 14 neurons (7 inputs, 5 inter-neurons and 2 outputs).

7.2 Analysis

A detailed analysis of an evolved robot performing the discrimination task was performed (note this is the single discrimination task rather than multiple). This analysis sheds light on the way

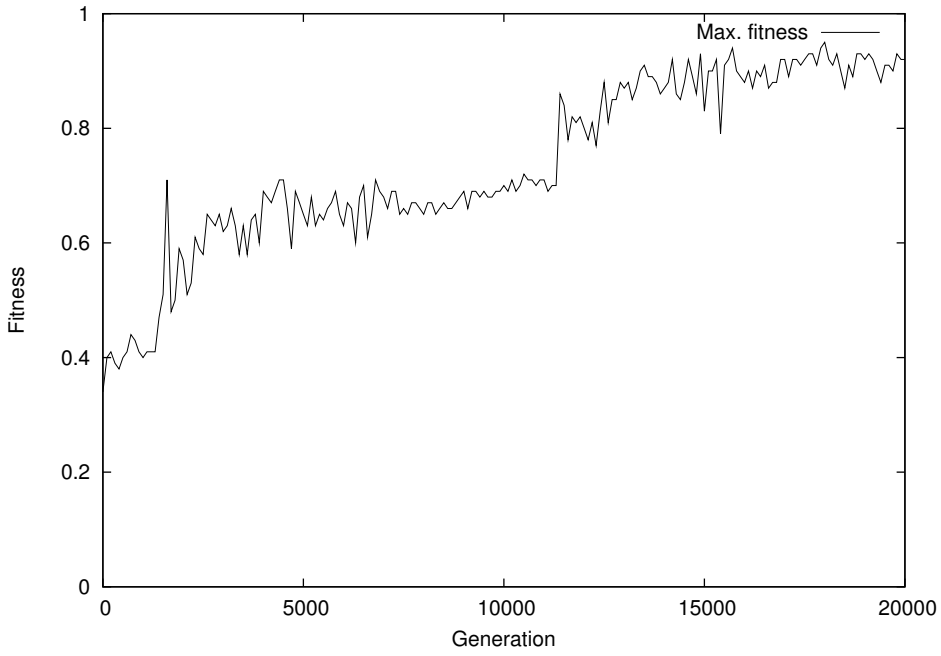

Figure 7.10: The fitness of the fittest individual in the population of robots with developmental neural network controllers evolving to perform the multiple discrimination task.

evolution interacts with the developmental process, and highlights some of the problems that must be overcome if complex controllers are to be generated through developmental systems.

7.2.1 Behaviour

Overview The evolved robot performs the behaviour successfully: over 10000 random trials the robot's mean fitness was 88% of the maximum possible.

Soon after the lifetime begins, the robot moves to the left regardless of whether a circle or a diamond is present in its environment. Later in its lifetime the robot's trajectory is affected by the shape of the object. If a circle is present, it doubles back, moving to the right at the appropriate speed to make contact with the object. Conversely, if a diamond is present, the robot continues its leftward motion, or comes to a halt at a good distance, thus producing the desired behaviour.

When a diamond is dropped to the far right of the robot, it avoids it by moving rapidly to the right, overshooting the object and avoiding it on its right-hand side.

Figure 7.12 shows the position of the robot plotted against time in two trials - firstly where a circle is falling and secondly where a diamond is falling. Here the circle and diamond fall at the same position and produce different behaviour in the robot. The trajectories are very similar for the first 36 seconds of simulated time, but they diverge after this moment. The 35-37 second region appears to be a critical moment for the robot.

Dependence on object diameter The details of the design of the robot and environment are designed to be a replication of Beer's (1996) discrimination experiments. These details include a difference in the maximum width of the circle and the diamond, since the former has a diameter of 30, whereas the latter has a side length of 30.

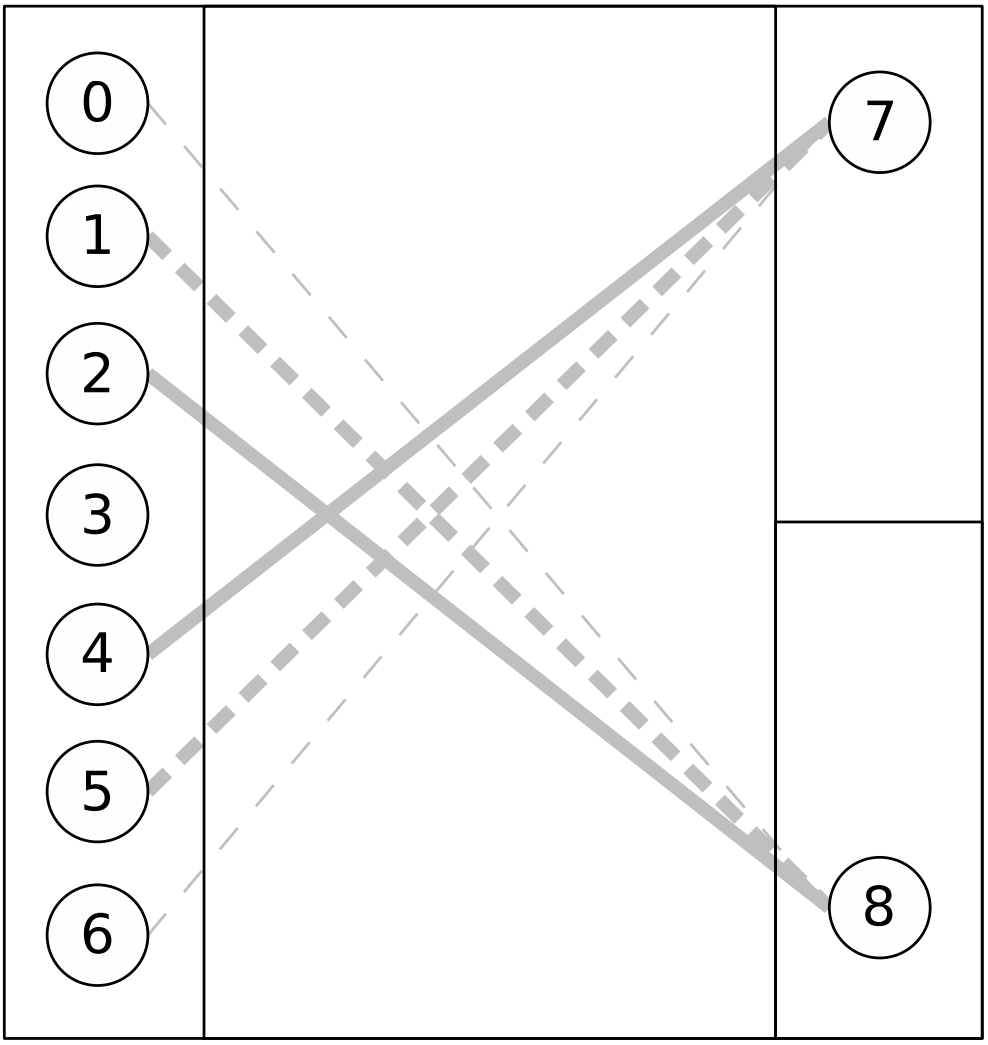

Figure 7.11: The controller of the fittest individual in the last generation of the best evolutionary run. This controller contains a very small number of neurons compared with other solutions to the same problem which typically contain 14 (Beer, 1996). Solid lines represent positive links and dotted lines represent negative links. A thicker line represents a stronger link.

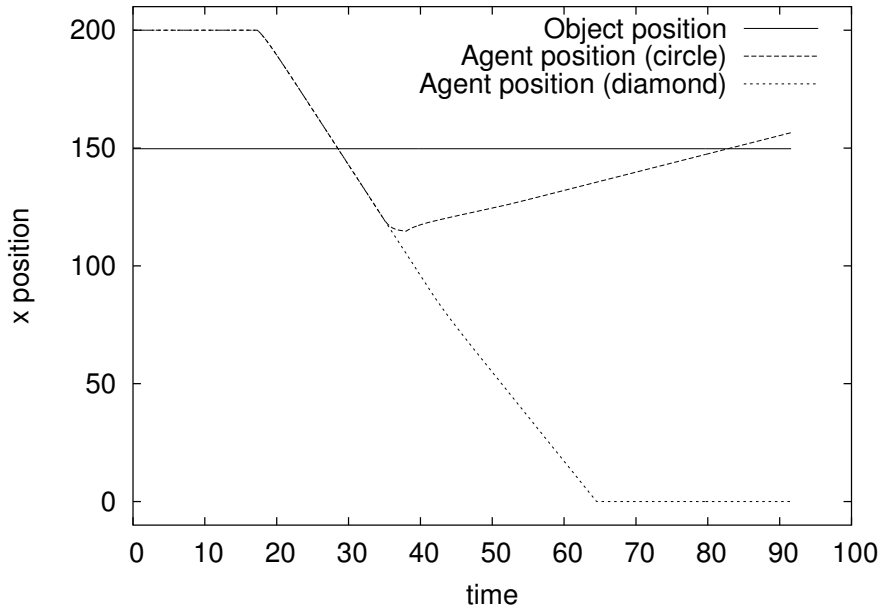

Figure 7.12: The horizontal position of the robot when either a circle or a diamond is present. The horizontal line shows the position of the object (circle or diamond). The behaviour is identical until a crucial point at around 30 seconds when the two situations diverge.

The robot performs 'scanning' behaviour with the initial leftward movement, and this together with the observations of Beer in examining CTRNNs engaging in the identical task (Beer, 2004) led to the hypothesis that the maximum width of the object may be a determining factor in the behaviour of the robot. This hypothesis was tested using the following two experiments.

First, the robot's behaviour was examined when it encountered circles of radius 18.4 units, giving them the same width as the maximum width of the diamond in the original experiment. Over 10000 trials, the mean fitness (where fitness was awarded for catching circles) was 7% of the maximum, meaning that the robot consistently avoided the large circles.

Second, the behaviour with smaller diamonds was studied. Diamonds with sides of length 18.4 were used, giving them the same maximum width as the width of the circle in the original experiment. Over 10000 trials, the mean fitness (where fitness was awarded for avoiding diamonds) was 30% of the maximum, meaning that the robot consistently caught the small diamonds. The higher fitness here is due to the fact that any slight imperfection in the catching behavior was rewarded, whereas if the robot avoids a circle by more than the maximal margin it receives a score of zero.

The initial hypothesis appears to be supported by these observations. It appears that the maximum width of the object is crucial to the discrimination process in this robot.

Critical period Observation of the behaviour and controller structure of the robot led to the hypothesis that there is a critical period early in its life that determines the behaviour it performs. This hypothesis was tested by presenting the robot with an object that abruptly changes shape during its lifetime, either from a circle to a diamond or vice-versa (this experiment was inspired by (Beer, 2004)). The time at which the switch occurred was varied, and the catching or avoiding performance was measured. The results are shown in figure 7.13, which shows that after a certain

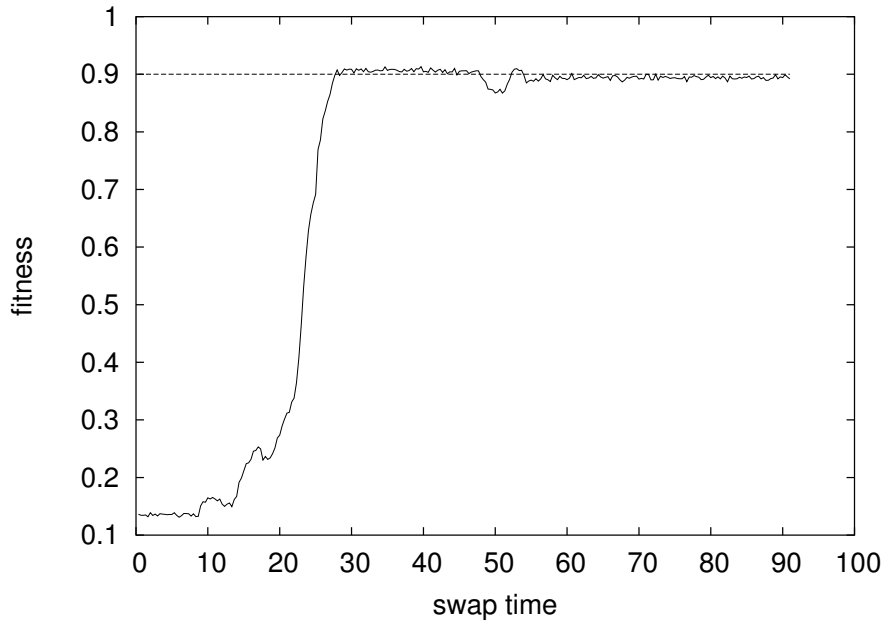

Figure 7.13: The mean fitness (over 10000 random trials) of the robot when the falling object is 'switched' part-way through its descent. The robot is awarded fitness for catching objects that begin as circles (later changing into diamonds) and for avoiding those that begin as diamonds (and become circles). The horizontal line indicates the mean fitness over 10000 random trials of the robot when no swap takes place. This graph shows that there is a critical development period for this robot.

amount of time the development of the robot is fixed into a specific trajectory even if the shape presented to it is changed.

Potential neural correlates of this behavioural observation are discussed in the next section.

7.2.2 Neural mechanisms

The robot has seven sensor neurons (numbered 0 to 6), and grows several other neurons during its lifetime, a maximum of two of which are functionally involved in the discrimination behaviour (neurons 7 and 8).

The robot's strategy is executed as follows: depending on the conditions leading up to the critical point (35-37secs) the robot becomes either a 'move-left' robot or a 'catch' robot. After this critical phase its behaviour is fixed.

'Default' behaviour: When it encounters no objects during its lifetime, the robot's 'default' behaviour is to move left for the first 30 simulated seconds, and then to move right for the remaining 60, ending up a little way to the right of its starting position. This behaviour is generated by two newly-grown neurons linked from various sensor neurons.

After 17 seconds neuron 7 is grown. Its parent neuron is neuron 6, which corresponds to the furthest right sensor. As shown in figure 7.14 (Left), this neuron grows into the influence region of motor 1. A positive link from neuron 6 to neuron 7 and a near-zero bias means that the firing rate of neuron 7 is high, and the motor is active, causing the robot to move leftwards.

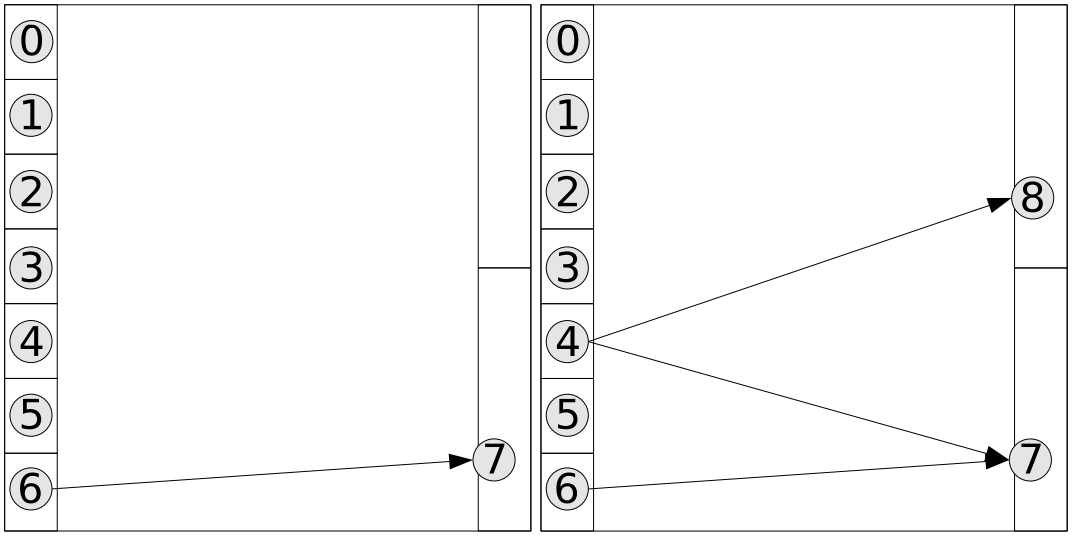

Figure 7.14: The robot's controller after 17 seconds (Left) and 31.5 seconds (Right) when neither a circle nor a diamond is present in the environment.

After 30 seconds, neuron 8 is grown into the influence region of motor 0. Its parent is neuron 4, and the link is a negative one. Neuron 8 has a positive bias, and this means its firing rate is at a medium level. Now both motors are active, but the robot is still moving slowly leftward since motor 1 is more strongly active.

Very soon after this (at 31.5 seconds), a new link is formed from neuron 4 to neuron 7. This is a negative link which quickly reduces the cell potential and hence the firing rate of neuron 7. This in turn reduces the activation of motor 1 and the robot moves back towards the right for the rest of its lifetime. This situation is shown in figure 7.14 (Right).

When a circle is present: When the robot encounters a circle, its behaviour is similar to that described above, except that the growth of neuron 8 is delayed until the critical region between 35 and 37 seconds. Neuron 8 still grows into the region of influence for motor 0, which, combined with a negative link from neuron 4 to neuron 7, causes the robot to move right and catch the circle. A further negative link from neuron 1 to neuron 7 strengthens this effect. The exact timings of growth and the firing rates of the new neurons determine the speed of rightward movement, causing the robot's trajectory to coincide with that of the circle when it reaches the bottom of the environment.

This behaviour is qualitatively similar at all of the different positions from which the circle may be dropped into the environment.

When a diamond is present: When the robot encounters a diamond, the behaviour is a little different. During the critical region, instead of growing a new neuron from neuron 4, the direction of growth is such that neuron 4 'tries' to grow a neuron at the same location as neuron 7, and instead a negative link is formed to this neuron. This, along with a further negative link from neuron 1 as in the previous section, slows the leftward movement (halting it in some circumstances), but since the region of influence of motor 0 does not contain any neurons, the robot does not move back to the right, but stays on the left of the object, avoiding it.

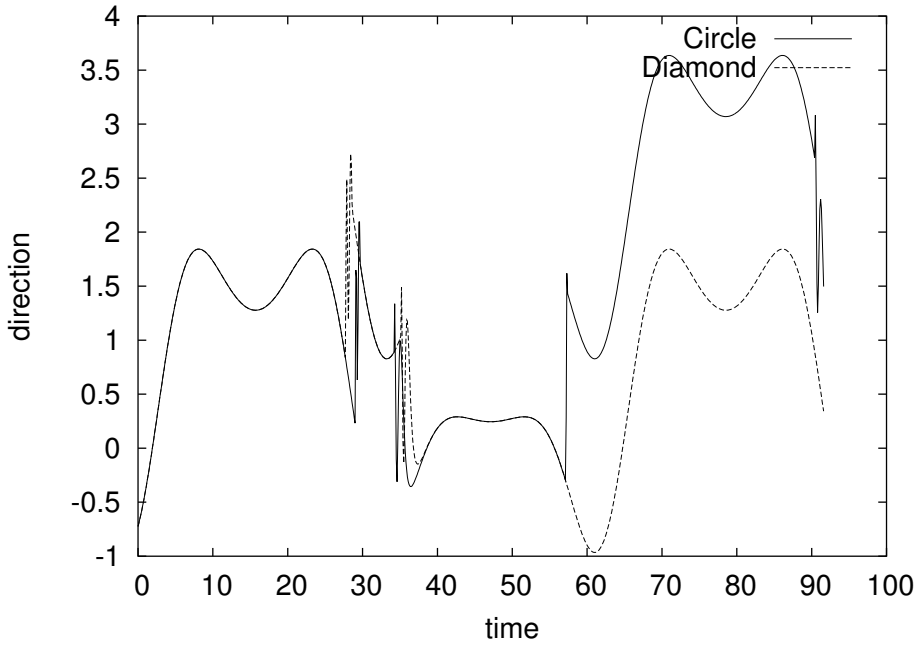

Figure 7.15: The potential direction in which a neuron will grow from neuron 4, if a neuron grows at this time. During the critical period, 35-37 seconds, the values are similar for the situation when a circle is present and when a diamond is present.

When the diamond falls at more than 30 units to the right of the robot, its behaviour is different. In these cases it avoids the diamond by passing underneath it and finishing to its right. This occurs because growth of neuron 7 is suppressed, preventing the initial leftward movement, and so when neuron 8 grows the robot's rightward movement carries it comfortably beyond the diamond.

7.2.3 Morphogenetic mechanisms

The crucial factor in determining the behaviour of the robot is what angle is chosen for the growth of neuron 8 from neuron 4. When a circle is present, this neuron is grown at an angle of -0.14 radians (slightly above the horizontal, towards the influence region of motor 0), whereas when a diamond is present it grows at 0.27 radians (below the horizontal, towards neuron 7 in the influence region of motor 1).

Figure 7.15 shows that the potential direction in which any neuron will be grown from neuron 4 is very similar in both cases during the crucial period. The variation in actual direction is caused by a slight difference in the timing of growth. This is controlled by the 'growth sum' variable, whose value in the circle and diamond cases is shown in figure 7.16.

Thus the method that evolution has found to differentiate between the two shapes is to allow the longer time of exposure to sensory input that is experienced when a diamond is present (due to its larger diameter) to cause a delay in the growth of a neuron. This delay means that the angle of growth is different, and the different location of the resulting neuron produces the observed behaviour.

The specific mechanism works by using the link between sensory input and cell potential. Figure 7.17 shows the cell potential y of neuron 4 in the cases when a circle or a diamond is

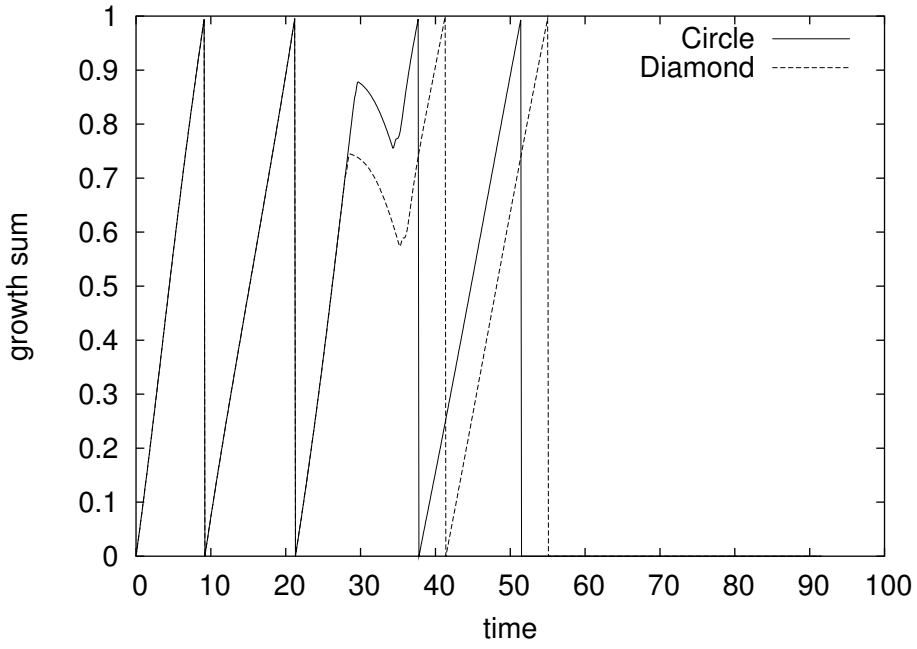

Figure 7.16: The growth sum of neuron 4 over the robot's lifetime. When the growth sum reaches 1, a new neuron or link is grown, and the growth sum is reset to zero. Note that with a diamond the crucial 3rd growth event is later than with a circle. This causes the growth direction to be different, and produces the behaviour described in the text.

present. It shows that the cell potential is at 1.6 (the value produced by this level of sensory input) for a longer period between 28 and 40 seconds when there is a diamond than when there is a circle. Figure 7.18 shows the amount by which the growth sum of neuron 4 is increased at each time step, for some values of its cell potential. The important values of cell potential are 0 (the cell potential is normally at 0) and 1.6 (during the critical period 28-40 seconds the cell potential jumps to 1.6). By looking up these values on figure 7.18 it may be seen that the increment when cell potential is 0 is 0.1 and when cell potential is 1.6 it is -0.02. Thus, the increase in growth sum is slowed by a longer period with cell potential at 1.6, which is caused by the longer sensory exposure associated with the presence of a diamond.

The development in the special case when a diamond falls to the far right of the robot's start position is slightly different. In this case the link from neuron 6 to neuron 7 is negative rather than positive, suppressing the activation of neuron 7, and preventing motor 1 from being activated. Now when neuron 8 is grown the activation of motor 0 is sufficient to move the robot quickly to the right, avoiding the object by passing under it. The difference in weight is again controlled by the timing of neuron growth - this time that of neuron 7. The weight of a new connection from neuron 6 is similar at any given moment for the situation when a diamond or circle are dropped at the same position, but in the diamond case the growth of neuron 7 is delayed until 30.6 seconds (when the weight is -0.67) whereas in the circle case it grows after 17.3 seconds (when the weight is 2.75).

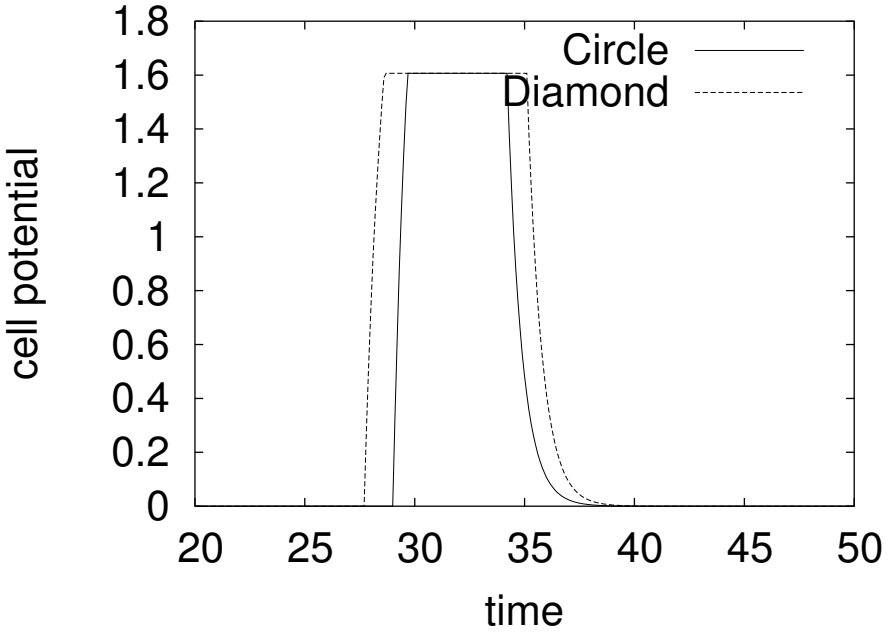

Figure 7.17: The cell potential of neuron 4 over the robot's lifetime when either a circle or a diamond is present in the environment.

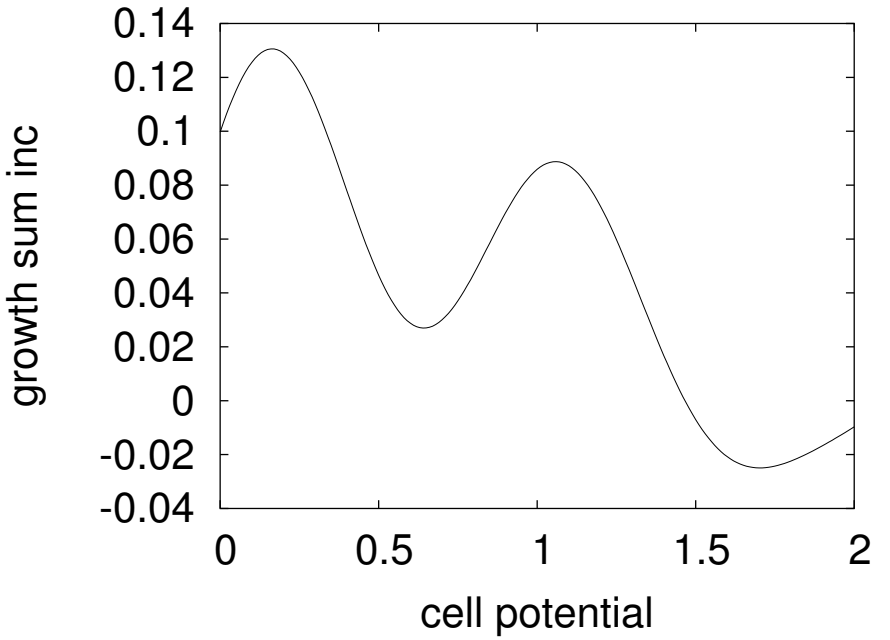

Figure 7.18: A function defined by the genotype giving the amount by which neuron 4's growth sum (the value that triggers new neuron growth when it reaches a threshold) is increased at a time step, against the cell potential of neuron 4. The values shown are for time fixed at 0, but in fact they are the same for all times.

7.2.4 Conclusions

The description above is dominated by events where neurons and synapses grow, rather than by the interactions of neuron activations and inter-neuron dynamics. The developmental system, which was conceived as a means of generating a CTRNN-like dynamical system that would control the robot, is actually being used as a direct control mechanism.

This result emphasises the tendency of evolution to use the most immediate, direct means of obtaining fitness available. If developmental systems are to be produced which operate as generators of lower-level dynamical systems (and gain the potential benefits of these two-tier systems), ways must be found either of preventing the developmental process from taking part in direct control, or of encouraging development to produce useful neural structures that will perform the control function more effectively that the developmental system itself.

It is interesting to note in the analysis that a crucial interaction in the production of the required behaviour is the effect of a kind of regulatory gene - the growth increment - on the action other genes which control the timing of growth. The growth sum is regulated by suppression of the growth increment in order to control the timing of the growth of a neuron. It has been observed that regulatory genes often have highly influential effects in developmental systems (Gould, 1977), and in this situation that appears to be the case.

7.3 Discussion

The controllers described in this chapter were designed to explore the potential benefits of development for designing useful robots and for increasing our understanding of the interactions between evolutionary and developmental processes. Since they combine many of the factors believed to be pivotal in developmental mechanisms in natural systems, such as spatial separation, chemical gradients, a single genotype for all neurons, scalability and growth in complexity over time, and since they have been shown to be amenable to evolution of successful behaviour in some benchmark experiments, they may prove useful in future research.

None of the tasks looked at here offer any real advantage to controllers which allow long-term change, so it might be expected that the additional complexity of developmental controllers would put them at a disadvantage, and some difficulty was encountered, but it was indeed shown to be possible to evolve the chemical-guided growth controllers to perform standard behaviours successfully. Even in the task which presented the most difficulty for the evolutionary process, the multiple discrimination task, when successful, the networks did find a very good solution which was subtle and effective and used only a small number of neurons.

After a description of some of the phenomena observed in section 7.3.1, sections 7.3.2, 7.3.3 and 7.3.4 look into some of the reasons for the difficulty encountered in the more complex experiments, and some of the lessons that have been learned.

7.3.1 Observations

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

Default paths In nature developmental processes are often understood by describing a 'default' path which development will follow unless disturbed by a perturbation. For example, in humans it is widely believed that,

> "Femaleness results from the absence of any masculinizing genetic factor or hormone acting during the critical period of differentiation." (Sizonenko, 2003)

Of course, this kind of situation may only be spoken of informally since the definitions of words like 'disturbed' and 'perturbation' in a developmental context are problematic (Oyama, 2000) (it is difficult to say from what the process has been disturbed).

Similarly to these natural situations, it was found that when no object was present in the environment, the development of the single-presentation discriminating robot continued to perform a 'default' behaviour similar in character to that performed in response to a circle. So it might be said that the presence of a diamond in the environment perturbed the default behaviour and pushed the robot into an alternative dynamics, causing it to avoid the object.

Chemical gradients The use of chemical gradients in the model provided the possibility of modular neural structures emerging, including collections of similar neurons in similar areas, and repeated patterns of neurons in different areas of the controller. Such phenomena were not frequently observed in the work described here, but when the chemical gradients were altered to be symmetrical, symmetrical neural structures were observed.

It is likely, if the shapes of chemical gradients themselves were under genetic control, and repeated patterns were allowed, that many more modular neural structures would be observed. This would realise one of the predicted advantages of using developmental over more traditional ones: modularity is expected to be a necessary property of controllers capable of highly complex behaviours.

Regulation of neuron growth The development of controllers in the experiments described did stabilise over time, and in some cases the 'energy' system of the neurons, loosely inspired by Nerve Growth Factor, was involved in that stabilisation, but some of the other dynamics made possible by regulation of neuron growth and differentiation are not possible in the system as described.

There is significant potential for extending the exploration of this area in future work. By continuing the analogy with Nerve Growth Factor in the model, allowing neurons to affect the development and growth of nearby neurons, many interesting developmental dynamics might be generated. For example, in a controller composed of two main types of neuron, if neurons were to emit chemicals which tended to cause nearby neurons to turn into the other type, a kind of grid could be generated, with alternating neuron types. A structure like this could be useful for many different tasks, including visual or auditory signal-processing. By allowing many different neuron types and feedback loops of chemical influence, it is possible that self-organising dynamics might be generated which produced complex and organised neuron structures suitable for a variety of different tasks.

Specialisation A potential advantage of the use of developmental controllers over CTRNNs is that development allows the robot to specialise according to the environment in which it finds itself. Of course, in the tasks studied there is no real need for specialisation, but it did occur in some cases. For example, as shown in figure 7.19 the same robot (in this case in the single-presentation

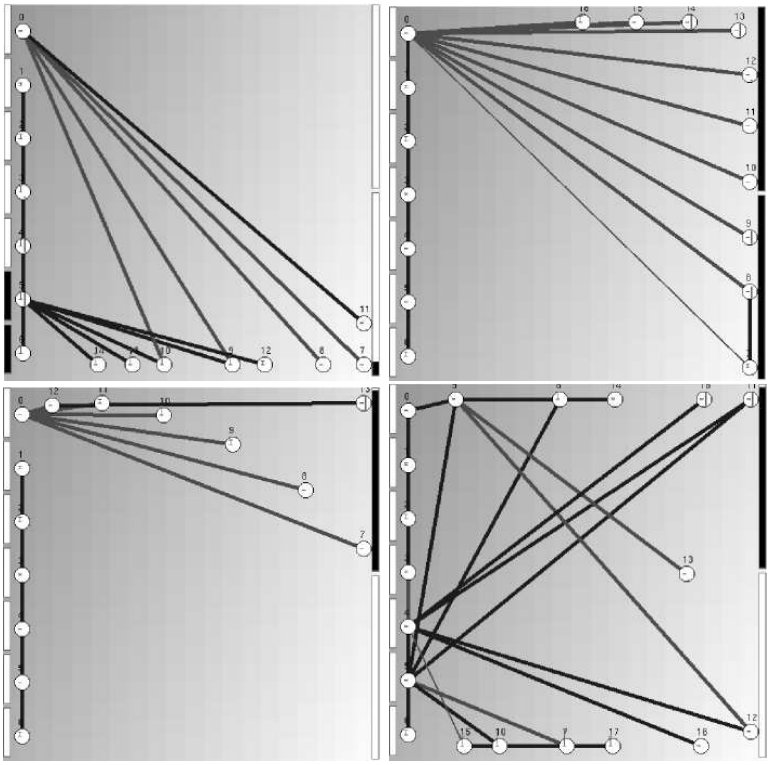

Figure 7.19: The controller structures grown by a single robot under different environmental conditions. The two structures at the top show the controller when circles are dropped at different horizontal positions. The two at the bottom show what happens when diamonds are dropped. Circles must be caught and diamonds avoided. Sensor neurons (with which the controller is seeded at the beginning of a lifetime) appear on the left, and motor neurons on the right.

discrimination task) developed different controller structures depending on the environmental conditions. The behaviours exhibited by this robot are shown in figure 7.20.

While the tendency to specialise may have negatively impacted the performance of robots 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.

Irreversibility As discussed in the previous section, the specialising behaviour of the analysed agent involved the selection of a particular path of dynamics, which then stabilised. Since many of the changes allowed in the developmental process in these controllers were irreversible, it is difficult for them to instantiate complex dynamics such as rhythmic or self-organising systems through the developmental system.

An interesting future extension of this work would be to allow the developmental processes to be reversed (allowing the death of neurons and synapses). This might allow for more flexible dynamics, and might provide a way for controllers to escape from over-specialised dynamics such as those which led the analysed controller to become fixed in one particular behaviour pattern.

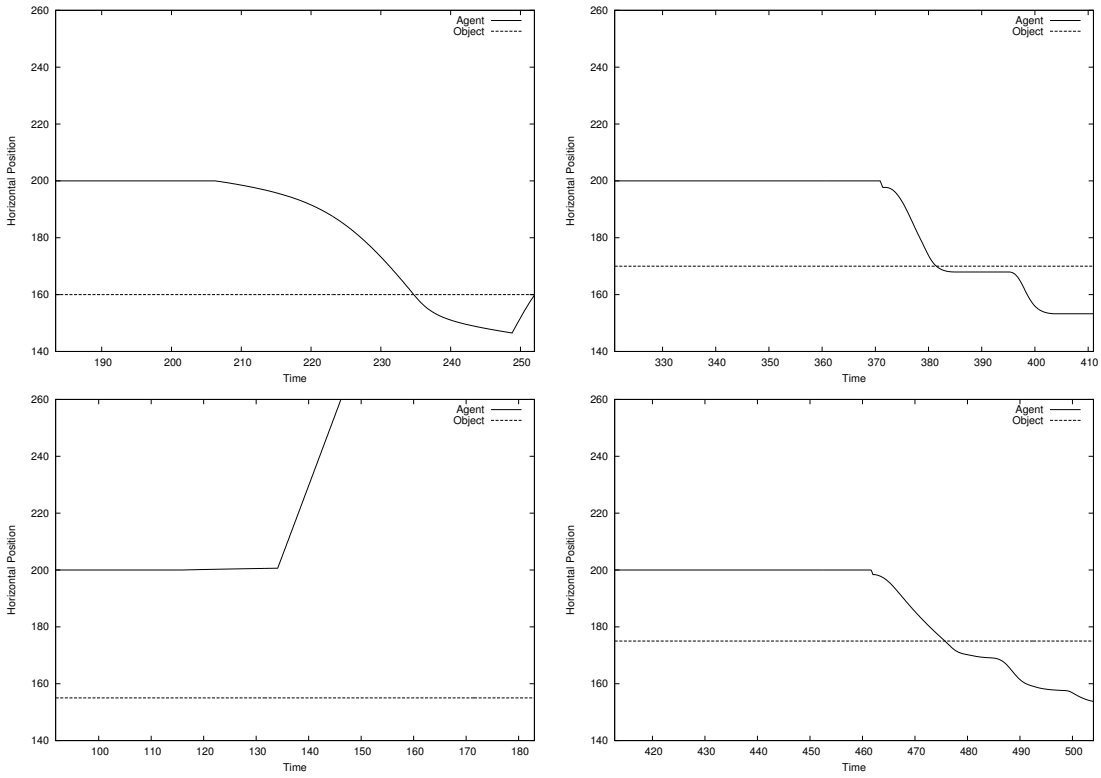

Figure 7.20: The behaviour of the robot with the controller illustrated in figure 7.19. The two paths at the top occur when circles are dropped at different horizontal positions. The two at the bottom show what happens when diamonds are dropped. The horizontal line shows the horizontal position of the falling object over time. The other line shows the position of the robot.

7.3.2 A two-tier process

The difference between performing a task once per lifetime and performing it in a stable way several times turned out to be much larger than expected. In a fixed-architecture controller, these situations are virtually identical (in a situation such as this where neurons had relatively small time constants), but in a developing controller they are completely different.

Performing a task once is a single-tier process of reacting to sensor input in a certain way. In contrast, performing a behaviour in a stable manner over several presentations is a two-tier process: initial development that stabilises, and produces the behaviour tier, which instantiates the behaviour itself. These two tiers occur over different timescales, and direct selection is applied to the lower behaviour tier, with the upper tier being indirectly selected only.

In order to investigate this problem further, evolution was used to evolve a simplified form of orientation behaviour: simply moving left when an object appears on the left and right when it appears on the right. It was found that even this behaviour was difficult to achieve in the system, and a correlation-based fitness function similar to that described in section 6.2.6 was required. In contrast, a hand-designed developmental controller for this task was designed quite easily which grew a very simple neural network at the beginning of the robot's lifetime.

Thus, the problem is not with the possibility of generating suitable controllers within the system, but with finding a smooth path towards those controllers. Evolution tends to find the simplest solution available, and a single-tier process is much simpler than a two-tier one. Using the 'developmental' (as intended) process as a form of direct control appears to be an isolated and attractive local optimum in the fitness landscape. There appears to be no smooth evolutionary pathway linking the one-tier approach to the two-tier one: this is unsurprising since they are of very different characters dynamically.

In real animals this 'bootstrapping' problem is not encountered since the development process is not an arbitrary system within which evolution must work, but an already-working system which must merely be adapted by evolution. In the multiple discrimination task the problem was overcome primarily by adding symmetry (this is discussed in section 7.3.4) and by introducing a strongly tailored fitness function. This fitness function essentially only rewarded behaviour that truly reflected the required solution, forcing evolution to search randomly until such a behaviour was found, before optimising that behaviour. As will be seen in part two and beyond, the choice of fitness function is crucial in overcoming the difficulties outlined in this section.

7.3.3 Navigating a rugged landscape

In addition to the inherent problems with a developmental system, some specific design features of the chemical-guided controllers produced important effects, including a seemingly rugged fitness landscape. Reducing the complexity of the mapping between genotype and behaviour by 'refusing' to use the potential two-tier process and reducing it to one tier allowed evolution to mitigate the effects of the instability of solutions in many areas of the fitness landscape.

Small changes in, for example the genotype mapping function which produces the angle of growth of a new neuron, can cause a radical change in network structure (for example a neuron that would have been linked to another pre-existing neuron now grows a new one instead) which can have a corresponding catastrophic effect on the robot's behaviour. Such catastrophic mutations are

common in many areas of the landscape, and may account for some of the difficulty experienced in evolving the required behaviour.

The ruggedness of the fitness landscape described above and the instability of networks under mutation mean that there are few stable paths towards networks of large numbers of highly-connected neurons. Such structures are rare in the fitness landscape anyway, but it appears they are also located in isolated spikes, rather than being part of a coherent area of the landscape.

Even the successfully evolved controllers are simple in structure and use small numbers of neurons.

The success of the more direct genotype mapping function is part of a more general learning point from this work: that the level of indirection between genotype and phenotype can become unmanageably large, and in such cases giving the genotype more direct control can be useful. Developmental systems are likely often to encounter this problem, and careful thought must be applied to reducing the complexity of the genotype-phenotype relationship to allow successful evolution.

In addition to increasing evolvability, simplifying the process of mapping from genotype to phenotype makes it easier to isolate features of the indirect encoding being studied. This is much more difficult if there are several levels of indirection involved.

The issues discussed above highlight a major learning point from this work: that the specific choices made in the design of a developmental network are crucial to its successful evolution. This is a surprising result since it might have been imagined that evolution would find the advantages and flexibilities inherent in any developmental system and exploit them to produce the required behaviours. In fact, it seems that with some designs of developmental system, the potential advantages of development are outweighed by the real disadvantages of much greater complexity, and less easily navigable fitness landscapes.

7.3.4 Generating structure from a blank slate

In the evolution of behaviours and in specific studies into the properties of the chemical-guided growth controllers, it was found to be surprisingly difficult to evolve useful neural structures, for example clusters of highly-connected neurons. A large amount of the evolutionary and developmental 'effort' seems to go into generating a basic structure of neurons connected in a useful way. In most neural network models, this basic structure is available 'for free' since it is pre-specified. In the brains of real animals, there is often no lack of raw materials to work with, so this constraint appears to be unnecessary as well as unhelpful.

The challenge is to make it easy to construct such a structure while still avoiding pre-specification. One approach is to provide the developmental process with 'raw material' in the form of neurons connected by synapses, but without any genetic input into the specifics of the structure.

A major factor in the eventual successful evolution of the required behaviour was the addition of symmetry to the controllers. This simple structural constraint actually makes the task significantly easier (since the task is symmetrical, that aspect of the problem is essentially solved for the robot), but it also helps to guide evolution down useful paths.

One of the familiar situations in attempts to evolve multiple discrimination was to find evolution stuck in a very poor local optimum of simply moving either left or right when either type of

object was detected. This meant that a proportion of robots were lucky enough to receive a good score. By constraining the controllers to be symmetrical, it becomes much harder not to react to the position of the object, which provides a route for evolution to follow towards reacting to its properties.

In general, it should not be underestimated how important symmetry can be for solving tasks like this, and the comparison between the symmetrical controllers of Beer (1996) and the asymmetrical ones described above is not entirely fair for this reason.

The importance of the need for pre-existing structure like connected neurons and symmetry drives home the point that the specifics of a developmental system are crucial to its evolvability, and to the forms evolved solutions take.

The next chapter describes how the conclusions above were applied in the design of a second piece of experimental work.

Chapter 8

Part 2 - Synaptic growth and pruning networks

8.1 Motivation

The system described in part one displays some limitations exposed by the difficulty experienced in evolving the chosen behaviours. Examination of the populations during evolution leads to the conclusion that some of that difficulty comes from the fact that the controllers start 'empty' (with few neurons and little pre-specified structure), and require the development process to 'fill' them with useful structure.

This is a bootstrapping problem in that until some kind of useful structure is reliably present in individuals, evolution is unable to adapt it to perform the required behaviours. In the controllers of part one any structure that is built is quite sparse and unstable to mutations, so there is little opportunity for evolution to modify the development process to optimise and innovate in the system.

This constraint is not one that is present in natural systems. Often, there is an abundance of structure to be worked with, and the challenge for development is to order it into useful forms. As explained in section 3.3, some researchers believe that brains undergo a period of 'exuberant growth' near the beginning of their development, and that the structures generated during that period are later refined as they develop.

Therefore, a reasonable next step from the work described in part one is to study controllers that are born with randomly-specified 'raw material' structure which may be refined. Controllers that work in this way are described in section 8.2.2.

Section 4.1.3 outlines some of the potential advantages which may be gained by the use of developmental controllers for robots. Experiments which explore whether some of these advantages are indeed gained by the chosen controllers are described in section 8.2.

The ongoing debate in the field of neuroscience between 'selectionists' and 'constructivists' is described in section 3.3. It concerns whether the development of brain structures takes place mainly through random exuberant growth and activity-dependent pruning (selectionism) or through activity-dependent growth and pruning (constructivism). The work described in this chapter has some relevance to this debate, particularly the experiment described in section 8.2.14, and some tentative conclusions are drawn in chapter 9.

The controllers of part 2 were tested in many different environments, and on many different tasks, and have been shown to be capable of generating successful behaviours in all the areas studied. This emphasises the potential utility of these controllers as general-purpose controllers capable of great flexibility. This flexibility means that they offer the potential to be useful in a large number of different areas involving development, being used to ask and answer questions about many different aspects of the interactions between development, evolution and behaviour.

8.1.1 From part one to part two

This work involves a model which is different from that used in part one in several ways. The chemical-guided controller adds a great deal of complexity to the CTRNN model, and this complexity makes it difficult not only to evolve complex behaviours, but also to analyse the behaviour of the model in meaningful ways. Thus it is desirable to reduce the level of complexity, and to replace some of it with better understood concepts such as the synaptic plasticity model introduced by Floreano and Mondada (1998). The controllers described in part 2 are simpler and more accessible to analysis than those described in part 1.

The spatial features of the controller have been removed, being replaced with a set of neurons that may be connected in any way. This change solves a significant problem found in part one - that high levels of connectivity were rare since it was difficult for neurons to 'find' other neurons to connect to. In the system described in section 8.2, connections are either on or off, and there is no need to seek out a neuron to which to connect. This allows for a very highly-connected network which is more likely to exhibit complex dynamics such as feedback mechanisms and self-organisation.

The most complex module of the system used in part one is the genotype mapping function. This introduces a lot of uncertainty, both as to the role it plays in the evolutionary process, and in terms of the significance it has for the difficulty of evolving the model. Although various alternative mappings were tried, and some insight was gained into what kinds of mapping are suitable, it was decided that direct specification of the required values in the genotype for the model in part two would remove one level of indirection from the design and analysis processes, which might be beneficial. The controllers of part 2 have a relatively direct genotype to phenotype mapping while preserving the indirection in the area of interest: development from an unordered towards an ordered state. This makes evolution easier, and focusses analysis on the phenomena of interest.

Due to the clear utility of imposing a symmetrical structure on the controllers in part one, the controllers designed for part two were designed from the beginning to be symmetrical.

In addition to the change to a new controller design, part two uses tasks of a different nature from those studied in part one. In order for the potential benefits of developmental controllers to be explored, the environment in which a robot acts must be complex enough to pose challenges which require adaptation and absorption of environmental information. Furthermore, the abilities of the robot itself must be complex and flexible enough to allow for different classes of action: in particular, it was felt that the one-dimensional movement of robots performing the orientation and discrimination tasks did not allow for enough possible outcomes to satisfy the need for several different classes of outcome. Thus, a two-dimensional environment is used in part two, which the

robot may explore and in which it may act with greater freedom.

The two controller types may be characterised as having certain advantages and disadvantages. The advantages of the chemical guided growth controllers are the large number of ways in which their design is inspired by real brains (such as the use of spatial location, growth from a small to a large network, the neuron regulation loosely inspired by Nerve Growth Factor, and a single genotype for all neurons) and their potential scalability to large numbers of neurons without a corresponding increase in genotype length. Their disadvantages relate to the difficulty found in evolving them to perform certain tasks, the uncertainty about the influence of different types of genotype mapping function, the irreversibility of many of the changes which occur during development, and the need to build complex dynamics from an initially very sparse system.

The disadvantages of the controllers of part 2 are that they do not share the scalability of the first controllers, requiring more genetically-controlled parameters for each neuron added, and the removal of several biologically-inspired features such as spatial locations and growth from a small to a large network. The advantages of these controllers are that they may be evolved successfully to perform many behaviours, they represent a simpler extension to CTRNNs (allowing for more complete analysis and introducing fewer assumptions and artefacts into the experiments) and, inspired by the situation in developing animals, they allow the adaptation during development of a complex system of dynamics from a less-ordered to a more-ordered state, rather than building complex dynamics from an empty controller. The experiments described in this chapter were performed with a relatively large simulation time step, making them inaccurate simulations of true CTRNNs. However, they retain useful properties of CTRNNs such as being smooth complex dynamical systems which may be shaped by the evolutionary process.

8.1.2 Comparison between controller types

Much of the work in part two involves exploring the open questions outlined in section 4.2 about the kinds of behaviour that may be generated by developmental controllers. In order to explore these questions, three different controller types are evolved using identical environments and evolutionary algorithms and the resulting evolved robots are studied.

The three controller types which are studied are CTRNN controllers with weighted inputs (‘fixed’ controllers), CTRNN controllers with plastic learning rules on the synapses (‘plastic’ controllers) and a new type of controller which is based on a CTRNN, with plastic synapses and additional rules for the growth and death of synapses (‘growth and pruning’ controllers). Both plastic and growth and pruning controllers have the explicit capacity for two-tier, structural change, but the growth and pruning controllers are capable of more radical changes in structure.

In all cases, the large simulation time step used means that the dynamics of these controllers do not accurately match those of true CTRNNs. This does not affect their useful properties of being smooth complex dynamical systems, shaped by the evolutionary process, and generally exhibiting short-term dynamics (relative to the longer-term changes seen in the plastic and growth and pruning controllers).

All of the controllers studied have the same number of neurons, which means that the more complex controllers have larger numbers of genetically-controlled parameters than the less complex ones. In this sense the comparison is thus ‘unfair’: more complex controllers may be neces-

sary for good performance of complex tasks. However, this goes to the heart of the question being asked: is the addition of this very specific kind of extra complexity beneficial for evolving better controllers for complex tasks?

In one experiment, larger fixed controllers with longer time constants were used, to allow for a more 'fair' comparison in order to address the specific question of whether particular mechanisms, either explicitly developmental or not, tend to produce more robust controllers through evolution.

All of the evolutionary runs were performed with identical mutation systems, as described in section 8.2.3, which might unfairly bias evolution towards one type of controller. However, the mutation rates have been tuned to produce good results for all the controllers, and in general when mutations rates are reasonable, this tends to affect mutation speed rather than the final fitness, and since most runs were left for long enough for a reasonable amount of confidence that the maximum fitness had been achieved, this is unlikely to have a significant effect on the final outcome.

8.2 Methods

The previous section describes the reasons behind the choices made in the design of the controllers and environments used in part two. This section describes their design in detail.

8.2.1 Robot and environment

The tasks used in part two are based in a two-dimensional simulated environment and involve controlling a wheeled robot.

The robot is circular, and navigates using two motors driving wheels on either side. Its direction is calculated as follows:

$$\dot{\theta} = \frac{r(F_l - F_r)}{20} \tag{8.1}$$

where θ is the angle in radians in which the robot is facing, $r \in [3:7]$ is the radius of the robot (randomly chosen at the beginning of each lifetime), and F_l and $F_r \in [0:0.5]$ are the forces exerted by each motor (proportional to the motor activation). The formula for calculating F_l and F_r follows:

$$F = 0.5\sum_{i=0}^{n} w_i y_i \tag{8.2}$$

where F is either F_l or F_r, the w_i are the weights of the synapses entering the motor and the y_i are the firing rates of the neurons or activations of sensors from which those synapses are connected. At each time step uniformly distributed noise in $[-0.05:0.05]$ is added to F and it is clipped to be within $[0:0.5]$.

Its speed is calculated as follows:

$$\dot{s} = -0.9s + (F_l + F_r) \tag{8.3}$$

These values are calculated using Euler integration with an integration step of 0.5.

The robot has several different types of sensor that were used in different experiments.

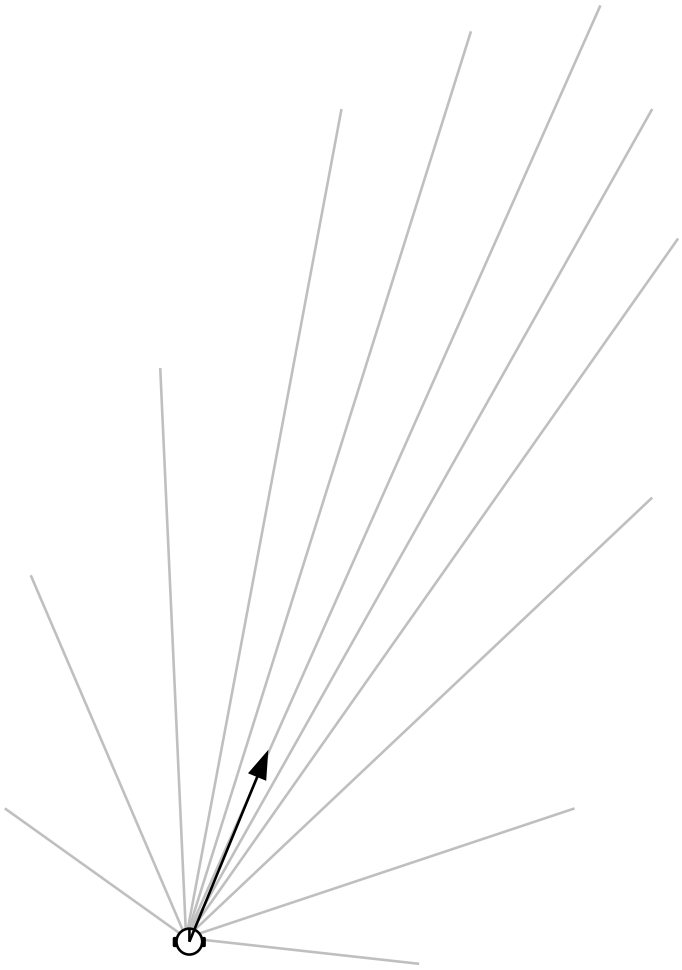

Figure 8.1: A single directional goal sensor. The sensor points at an angle of $3\pi/8$ from the robot's direction of facing, and its activity depends on the location of the goal. If the goal is close and is placed in the direction of facing of the sensor, the activation is high. If it is far away or is in a different direction, the activation is low.

A directional goal sensor is illustrated in figure 8.1. When the robot has these sensors, it has two of them, pointing in directions $\frac{-3\pi}{8}$ and $\frac{3\pi}{8}$ relative to the direction of facing of the robot. The activity of one of these sensors is inversely proportional both to the angular distance between the sensor's direction and the direction to the goal, and the distance of the goal from the robot. Activation is calculated as follows:

$$a = ang.dist \tag{8.4}$$

$$\text{where} \quad ang = 1 - \frac{A}{\pi/2} \quad \text{and} \quad dist = 1 - \frac{D}{L} \tag{8.5}$$

where A is the absolute angular difference between the direction of facing of the sensor and the line between the sensor and the goal (clipped to be $\leq \frac{\pi}{2}$), $L = 200$ is the range of the sensor, and D is the Euclidean distance between the sensor and the goal (clipped to be $\leq L$). It should be noted

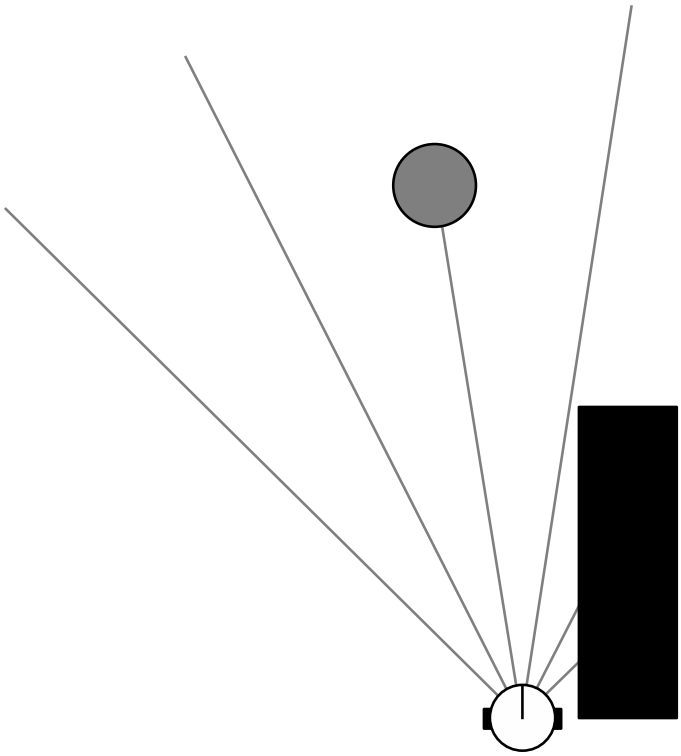

Figure 8.2: The robot's ray distance sensors. Activation of each sensor is inversely proportional to the distance to the first point of intersection of a ray with an object in the environment.

that these sensors were only used in the phototaxis experiments, and were not present in other experiments.

Distance sensors similar to those used in part one (section 6.2.1) were used, whose activation is proportional to how close to the robot the sensor beam is intersected. The formula for activation of one of these sensors follows:

$$S = 10\left(1 - \frac{d}{L}\right) \tag{8.6}$$

where d is the distance to the closest point of intersection of an object with this sensor, clipped to be $\leq L$ and L is the length of the sensor, equal to either 100 or 200. Thus, $S \in [0:10]$.

Also available are signal sensors, which have a constant activation of 1.0, except when the signal is turned on, when they have a constant activation of 9.0.

All sensor activations are subject to uniformly distributed noise in $[-1:1]$ and are clipped to be within $[0:10]$.

The environment contains structures such as corridor walls that are formed of lines that may be detected by ray distance sensors, and that cause a collision if one of the four compass points on the robot's circular body crosses these lines. If such a collision occurs, the robot's velocity is altered as follows:

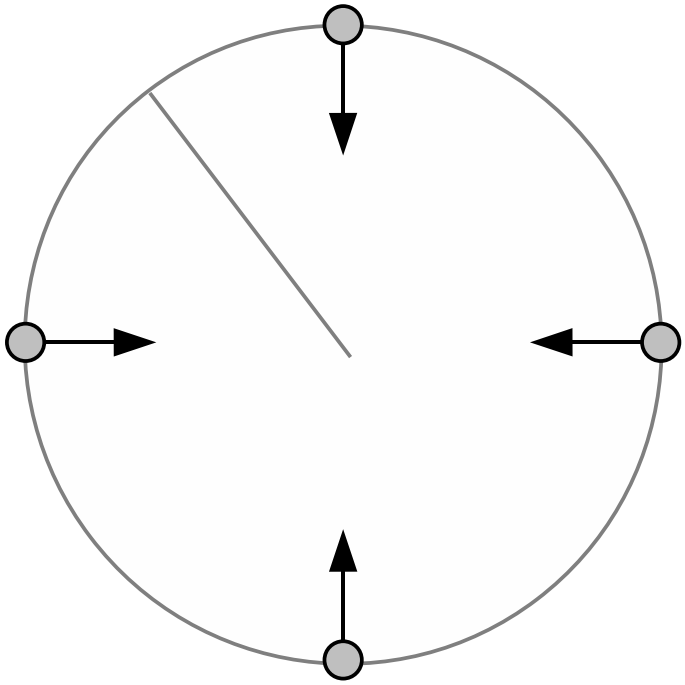

Figure 8.3: If one of the compass points on the agent's body collides with a wall it receives an impulse which pushes it away at half of its velocity in the relevant dimension away from the point of collision.

$$\begin{array}{rcrl} v_x & = & 0.5|v_x| & \text{if the collision is on the left} \\ v_x & = & -0.5|v_x| & \text{if the collision is on the right} \\ v_y & = & 0.5|v_y| & \text{if the collision is on the bottom} \\ v_y & = & -0.5|v_y| & \text{if the collision is on the top} \end{array} \tag{8.7}$$

where v_x and v_y are the robot's speed in the horizontal and vertical directions respectively.

This system offers rudimentary collision detection that works reasonably well in practice without costing a great deal in computational complexity. It is illustrated in figure 8.3.

8.2.2 Controller

The 'growth and pruning' controller is designed to satisfy the motivations outlined in section 8.1. As in part one this controller is an extension of the CTRNN design to allow structural change to occur during the lifetime of the robot.

A slightly different network structure was used with the difference from part one that sensors and motors are seen as separate nodes in the network, with weighted synapses connecting them to and from the neurons. This allows each sensor to influence more than one neuron, and each neuron to be influenced by more than one sensor. The equation for the activation of a neuron is a slight variation on that given in equation 5.1:

$$\tau_i \dot{y}_i = -y_i + \sum_j w_{ji} z_j \tag{8.8}$$

where here the z_j refer to the firing rates of neurons connected to neuron i and the activations of sensors connected to this neuron by weighted inputs. $\sum_j$ denotes the sum over all neurons and sensors j connected to neuron i, $y_i \in [-8:8]$ is the cell potential, $\tau_i \in [1:2]$ the time constant, and $w_{ji} \in [-5:5]$ is the weight of the connection from neuron or sensor j to neuron i. When z_j refers to a neuron it is calculated as shown in equation 5.2. When z_j in equation 8.8 refers to a sensor it equals the activation of that sensor divided by 10.

These values are calculated using Euler integration with an integration step of 0.5. Since a CTRNN simulated using this integration step diverges quickly from the behaviour of one with a smaller integration step, the CTRNNs described here should be regarded as dynamical systems under genetic control, rather than accurate simulations of CTRNN dynamics. Experiments were performed with smaller integration steps, and the results were found to be very similar to those performed here. This integration step was chosen due to the time constraints involved in running large numbers of repetitions of experiments under many different sets of conditions.

At each step the change in neuron cell potential is subject to uniform noise in $[-0.01:0.01]$.

In part two, following from the benefits of symmetric controllers found in part one, the controllers are constrained to be symmetric. This constraint applies to the genetically-set properties of the neurons and synapses, but the development process is free to follow paths leading to non-symmetrical configurations (and, since sensory input and noise are rarely symmetrical, it often does).

Activations and firing rates of the neurons in the controllers change according to the above equation. In addition to the processes of the CTRNN-based model, further processes act, producing weight change and growth and death of synapses. In some cases the network is initialised with only some of its synapses alive.

The networks used contain 6 fully interconnected neurons, with several input sensor units fully connected to the neurons and two output motor units fully connected from the neurons. The term 'fully connected' means that potential synapses, either living or dead, exist between all the units. Dead synapses do not pass any activation from the presynaptic to the postsynaptic neuron.

Weight change The synaptic weights in the neural network change according to the following rule:

$$\dot{w} = \alpha(Ax + By + Cxy) \tag{8.9}$$

where A, B and C are genetically-set values, x is the firing rate of the presynaptic neuron, y is the firing rate of the postsynaptic neuron, and α is a directional damping factor (Di Paolo, 2003) defined as follows:

$$\alpha = \begin{cases} 1 - w/8 & \text{if } w \geq 0 \\ w/8 & \text{if } w < 0 \end{cases} \tag{8.10}$$

This damping factor constrains the weight to be within $[-8:8]$ and tends to produce weight distributions which are widely spread over the possible range rather than collected at its boundaries.

Synapse growth and death In addition to the A, B and C values mentioned above, each synapse has 8 genetically-determined values XG_1, XG_2, $YG1$, YG_2, XD_1, XD_2, YD_1 and YD_2 that define the ranges of pre- and postsynaptic neuron firing rates that cause growth and death. Each of these values is in the range [-8:8]. Thus, the probability of a given synapse growing between two neurons (when one is not already present) during a simulated second is:

$$P(grow) = \begin{cases} \lambda & \text{if } x \in [XG_1 : XG_2] \text{ and } y \in [YG_1 : YG_2] \\ 0 & \text{otherwise} \end{cases} \tag{8.11}$$

and the probability of a living synapse dying during a second is:

$$P(die) = \begin{cases} \mu & \text{if } x \in [XD_1 : XD_2] \text{ and } y \in [YD_1 : YD_2] \\ 0 & \text{otherwise} \end{cases} \tag{8.12}$$

where x and y, as above, are the pre- and postsynaptic neural firing rates respectively. The values λ and μ are fixed for the course of each evolutionary run, but were varied between runs to investigate the utility of synapse growth and death for evolving behaviour.

Network initialisation At the beginning of the robot's lifetime, the network is initialised in a state designed to be analogous to the state of a real brain after a 'period of exuberant growth' at the beginning of its lifetime. Thus, initially, each pair of neurons N_1, N_2 has a probability of γ of being connected by a synapse $N_1 \rightarrow N_2$ and a probability of γ of being connected by a synapse $N_2 \rightarrow N_1$. Similarly, each potential link between a sensor input and a neuron and between a neuron and an motor output has a probability of γ of actually containing a live synapse. The value of γ was fixed within any evolutionary run but was varied in different experiments between 50% and 100%. In the following chapter, evolutionary runs performed with γ set to 90% are labelled as 'Growth and Pruning 90' while runs with γ set to 100% are labelled as 'Growth and Pruning 100,' and others are labelled similarly.

All synapses begin with a random weight in the range $[-8 : 8]$. Therefore, the developmental process cannot rely on either the existence of any specific synapse or its weight at the beginning of the robot's lifetime. Evolution must use the growth and death rules with the weight change rules to produce the required behaviour.

The experiments involve comparisons with fixed and plastic neural network controllers. The fixed controllers, which have a maximum of one input per neuron, use the above CTRNN equation, with fixed weight values and all connections alive. The plastic controllers are identical to the full growth and pruning controllers except that all synapses are alive and there is no possibility of growth or death of synapses.

In one experiment 2 further controller types were used, referred to as 'medium fixed' and 'large fixed.' These controllers are similar to the fixed controllers, except consisting of 18 and 28 neurons respectively, instead of 6 (as in the normal fixed controllers), and allowing for longer time constant ranges. The number of neurons was chosen in order to result in a genotype which is similar in length to that of the growth and pruning controllers, to allow comparison of different ways of using a similar-length genotype. The time constants of neurons in these controllers were given by the following formula:

$$\tau = 10^{X log_{10}(150)} \tag{8.13}$$

where X is a genetically-set value in $[0:1]$. Thus, $\tau \in [1:150]$ and a randomly chosen value of X is more likely to produce a small τ than large, allowing for fine tuning of short time constants as well as potentially very long time constants. The length of experiments in which this controller was used was 150 time units, meaning that the temporal range of these controllers was allowed to span the full lifetime of the robot, as is the case with semi-permanent changes in growth and pruning controllers such as the growth or death of a synapse.

8.2.3 Genetic algorithm

A generational, asexual genetic algorithm using rank selection with elitism (10% elitist fraction) on a population size of 30 is used, with fixed-length real-valued genotypes.

Genotypes for the growth and pruning controllers contain values for the bias and time constant of each neuron as well as 11 values per synapse, encoding the 3 coefficients used in the learning rule and the 8 range boundaries used in the synapse growth and death rules (see section 8.2.2 for details). This gives a genotype of length 534.

In fixed controllers genotypes simply encode the bias and time constant of each neuron and the weight of each connection. This gives a genotype length of 108 for the standard 6-neuron fixed controllers with, and 560 for the large 28-neurons ones.

In plastic controllers genotypes encode the bias and time constant of each neuron and 3 learning rule coefficients (all connections are alive in both cases). This gives a genotype length of 300.

Mutations are performed treating the real-valued genotypes as homogeneous vectors. A mutation consists of the addition of 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 0.5 (Beer, 1996), followed by a randomisation step where each value may be randomised between its maximum and minimum values with probability 0.6%.

8.2.4 Phototaxis task

The first task to which these controllers were applied was the phototaxis task. This task involves a robot with two directional goal sensors. It is required to approach the goal and stay as close as possible to it.

The robot receives fitness for being close to the goal. The goal is placed at a distance in the range $[80:120]$ away from the robot at a random angle in $[0:2\pi)$ at the beginning of its lifetime. If the robot moves to within 15 units of the goal, the goal is re-placed relative to the robot in the same way. The trial ends when the time limit, chosen randomly from the range $[200:400]$ simulated seconds, is up.

Fitness is awarded as follows:

$$f = BO + (1-m^2)\int_0^T \left(1 - \frac{d}{D}\right) dt \tag{8.14}$$

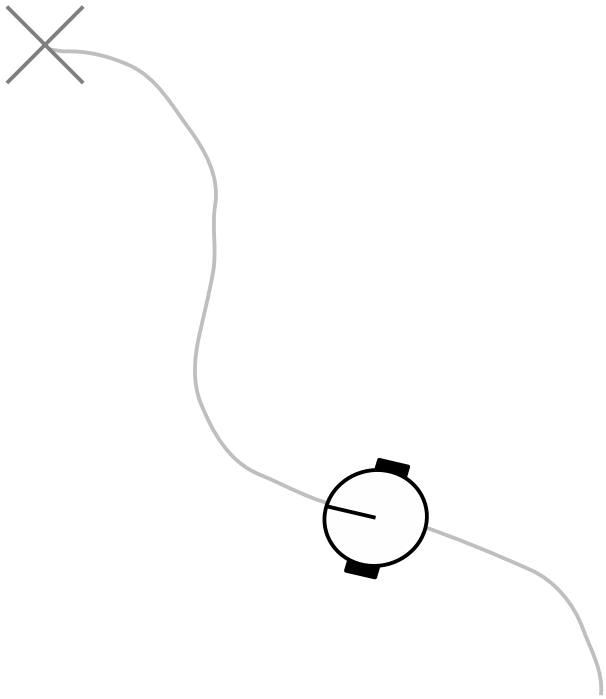

Figure 8.4: Phototaxis task: the robot must approach the goal using its two directional goal sensors.

where d is the instantaneous distance between the robot and the goal and D is the initial distance, and T is the length of the lifetime. The integral is approximated in the simulation using a discrete sum over all time steps. BO is a bonus score for touching the goal (and causing it to move):

$$BO = \frac{500n}{T} \tag{8.15}$$

where n is the number of times the robot hit the goal during the lifetime.

m measures the integrated difference between motor activations, encouraging the robot not to move in spirals (Di Paolo, 2003).

$$m = \frac{0.125}{T} \int_0^T (m_l - m_r)\, dt \tag{8.16}$$

where m_l is the instantaneous activation of the left motor and m_r is that of the right motor. The integral is approximated in the simulation using a discrete sum over all time steps.

The fitness of the robot for a trial consisting of 5 lifetimes is given by:

$$F_l = 0.5 f_m + 0.5 f_w \tag{8.17}$$

where f_m is the mean fitness over 5 lifetimes and f_w is the worst fitness achieved.

This task was evolved for growth and pruning 50, plastic and fixed controllers, as well as the special growth and pruning 95* controllers explained in section 8.2.12. The values of λ and μ (equations 8.11 and 8.12) were both set to 0.2. Each evolutionary run was allowed to proceed for 1000 generations, and 20 evolutionary runs were performed for every controller type.

This experiment is designed to shed light on open question 1 of section 4.2.

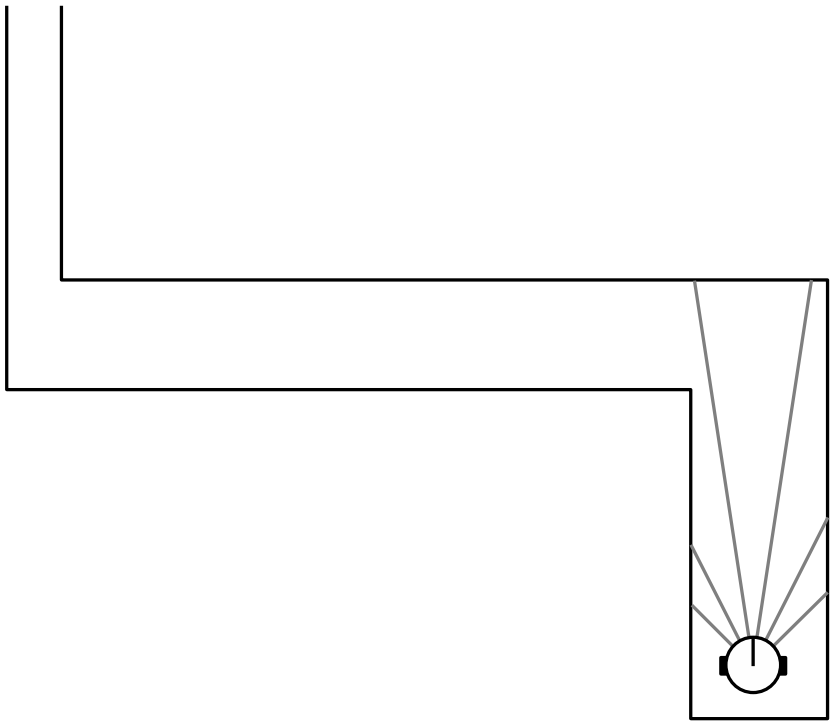

Figure 8.5: Corridor following task: the robot must navigate towards the end of the corridor without hitting the walls.

8.2.5 Corridor following task

The corridor following task is somewhat more complex than phototaxis but still involves mainly reactive control. The robot begins its lifetime at the end of a corridor and is required to move along it, avoiding coming into contact with the walls. The corridor is made up of 3 segments along which the robot must travel upwards, leftwards and upwards respectively. This scenario is illustrated in figure 8.5.

For this task the robot is equipped with an array of 6 ray distance sensors equally spaced over an angle of $\frac{\pi}{2}$, of length 200. Each corridor segment is of length $\in [50 : 250]$ and width $\in [20 : 40]$ chosen randomly, independently for each segment, at the beginning of the robot's lifetime. The robot begins its lifetime facing a random direction within $[-\pi : \pi)$ of the upward direction at a position whose x and y co-ordinates are within $[-5 : 5]$ of the point half of the corridor width above the bottom of the first corridor segment, and half of the corridor width from each side.

If the robot's circular body comes into contact with the corridor wall it undergoes a collision as described in section 8.2.1.

Fitness is awarded by setting a series of invisible goals at each corner of the corridor. At the beginning of the robot's lifetime a goal (which is not detectable by the robot's sensors) is placed at the centre of the corridor on the first corner. The robot is awarded fitness for approaching this goal, and if it comes within 30 distance units the robot receives a fitness bonus and the goal is moved to the next corner. The same process is applied to the second corner, and finally the goal is placed at the end of the corridor, where it stays for the remainder of the robot's lifetime.

A fitness penalty is applied if the robot hits a corridor wall. Its final fitness is multiplied by the

following value:

$$P = \frac{1-h}{500} \tag{8.18}$$

where h is the number of collision events that occurred during the robot's lifetime, capped to be $<= 500$. Thus if the robot hits a wall more than 500 times in its lifetime it scores zero, and otherwise its fitness is reduced for each collision that occurs.

In order to deal with the different dynamics of evolving to approach the first goal verses evolving to continue from that goal along the corridor, the fitness function is divided into two parts. If the robot never reaches the first goal, its lifetime lasts for 300 units of time, and its fitness is given by:

$$F = 0.01P \int_0^T \left(1 - \frac{d}{D}\right) dt \tag{8.19}$$

where $T = 300$ is the length of the lifetime, d is the final distance of the robot from the first goal and D is the initial distance of the robot from the first goal. The integral is approximated in the simulation using a discrete sum over all time steps.

However, if the robot does reach the first goal, its lifetime lasts for 200 time units from the moment of reaching that goal, and its fitness is calculated as follows:

$$F = P\left(BO + \int_{t_0}^{t_1} \left(1 - \frac{d}{D}\right) dt\right) \tag{8.20}$$

where t_0 is the time when the first goal was reached, $t_1 = t_0 + 200$ is the end of the lifetime, d is the distance of the robot from the current goal, D is the initial distance from the goal when the goal was last moved, and BO is the bonus fitness for reaching goals:

$$BO = \frac{100n}{T} \tag{8.21}$$

where n is the number of times the robot reached a goal during the lifetime (including reaching the first goal), and T is the full length of the lifetime, including the time before the first goal was reached. The integral is approximated in the simulation using a discrete sum over all time steps.

The fitness of the robot over a trial consisting of 4 lifetimes is simply the mean of each of the 4 lifetime fitnesses achieved.

This task was evolved for growth and pruning 90 and 100, plastic and fixed controllers. The values of λ and μ (equations 8.11 and 8.12) were both set to 0.2. Each evolutionary run was allowed to proceed for 1000 generations, and 10 evolutionary runs were performed for every controller type.

This experiment is designed to shed light on open question 1 of section 4.2.

8.2.6 Predictable change task

As discussed in section 4.1.3, developmental processes may make it easier to design robots that can adapt to predictable changes in their environment, morphology or required behaviour. This task requires the robot radically to change its behaviour during its lifetime, and tests whether developmental controllers may be evolved to undergo such a change successfully.

The task involves following a corridor for the first portion of the robot's life, and then when the end of the corridor is reached, the robot must change its behaviour from corridor-following to begin to approach a goal. In this case (in contrast to the phototaxis task) the robot can only sense the goal using its ray distance sensors. The goal appears as a small solid block which registers in the robot's sensors in the same way as the walls of the corridor. Thus the robot must change the nature of its behaviour from avoiding walls to approaching blocks when it detects that it has reached the end of the corridor.

The dimensions of the corridor are the same as in the previous section, but the robot's lifetime is extended by 600 time units when it reaches the first goal (i.e. $t_1 = t_0 + 600$ in equation 8.20), rather than 300 as in that section. When the goal at the top of the last corridor section is reached (the robot comes with 30 units of it), the goal moves to be within a square block whose side length is taken from a uniform distribution $\in [5:15]$. The block is located with x co-ordinate $\in [-100:100]$ from the robot's x co-ordinate and y co-ordinate $\in [25:225]$ from the robot's y co-ordinate (i.e. at least 25 units above the end of the corridor). When this goal is reached, the walls of the corridor are removed from the environment, and the goal is re-placed with x and y co-ordinates within $[-100:100]$ of the robot's position, and the corridor walls are removed from the environment. Each time the goal is reached, it is re-placed in the same way. This task is illustrated in figure 8.6.

If the robot does not reach the first goal (at the end of the first corridor segment) it receives fitness as in equation 8.19. If it does reach the goal, its fitness is calculated as in equation 8.20, using a larger fitness bonus:

$$BO = \frac{250n}{T} \tag{8.22}$$

where n is the number of times the robot reached a goal during the lifetime (including reaching the first goal), and T is the full length of the lifetime, including the time before the first goal was reached.

The fitness bonus is awarded both for reaching invisible goals within the corridor and for reaching the visible goals after the end of the corridor is reached. The final fitness of a robot is its mean fitness over 4 lifetimes.

20 evolutionary runs were performed for the growth and pruning 100 controllers, 20 for the plastic controllers, and 20 for fixed controllers. The values of λ and μ (equations 8.11 and 8.12) were both set to 0.2. Each run consisted of 2000 generations.

One additional set of 20 runs was performed, which featured the growth and pruning controllers, but under selection pressure for change to occur in the controller during the crucial period when the robot emerges from the end of the corridor (these controllers were labelled growth and pruning 100*). Fitness was awarded to these robots as follows:

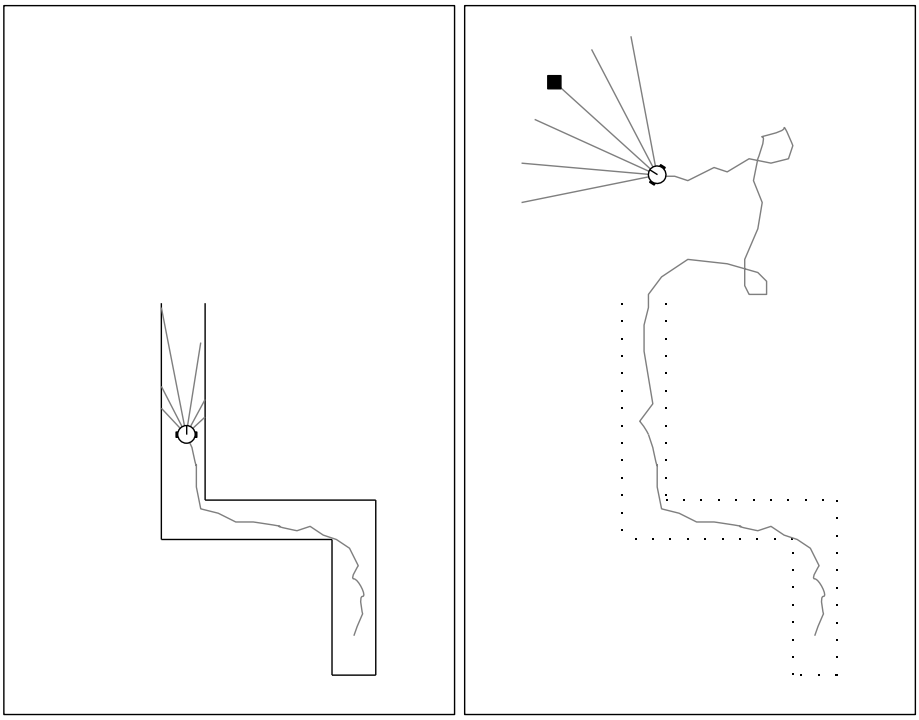

Figure 8.6: Predictable change task: The diagram on the left shows the robot while it is navigating along the corridor. The diagram on the right shows the robot approaching the goal. After the first goal has been found, the corridor walls disappear.

$$F = CP\left(BO + \int_{t_0}^{t_1}\left(1 - \frac{d}{D}\right)dt\right) \tag{8.23}$$

where all the variables are the same as those in equation 8.20, with the addition of the C, which is defined as:

$$C = 0.5 + \frac{changed}{total} \tag{8.24}$$

where *changed* is the number of synapses which have changed state (from alive to dead or from dead to alive) between the moment when the robot reaches the second goal and the end of its lifetime, and $total$ is the number of synapses in the controller of the robot. *changed* is capped to be $\leq total/2$, which means that $C \in [0:1]$. The integral is approximated in the simulation using a discrete sum over all time steps.

These additional runs were performed to investigate whether the additional selection for change in these controllers can guide evolution toward better solutions, but encouraging it to separate the roles of synapses to take part in either the corridor following or the phototaxis task, rather than producing a single unified controller which performs both tasks.

This experiment is designed to shed light on open questions 1, 2, 3 and 4 of section 4.2.

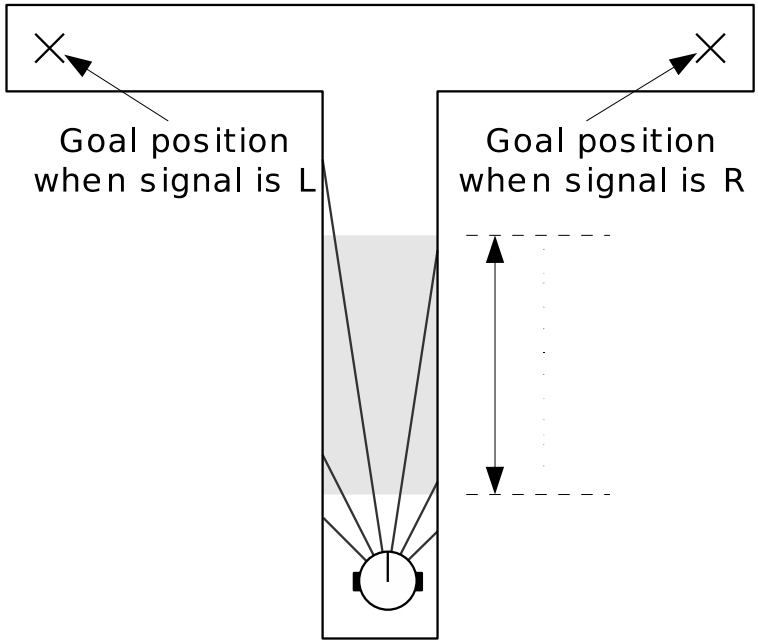

Figure 8.7: T maze task: The robot must go the end of the corridor and turn either left or right according to the signal it receives. The signal is active when the centre of the robot is within the grey area, above 25% of the corridor length and below 75%.

8.2.7 T maze task

The T maze task (introduced by Jakobi (1997)) provides a basis for several more complex tasks. It is of a higher complexity than phototaxis and corridor following, since it is relatively non-reactive. The robot begins at the end of a corridor and receives a signal as it moves up the corridor. When it reaches the top it must turn left or right depending on the signal it received earlier in its lifetime. Figure 8.7 illustrates this situation.

The corridor dimensions are randomly chosen from uniform distributions. The vertical corridor is of length $\in [140:160]$ and width $\in [25:35]$ and the horizontal corridor is of length $\in [180:220]$ and width $\in [25:35]$. The horizontal corridor is centred over the vertical one, so the left and right sections are of identical length.

The robot is equipped with 6 ray distance sensors of length 100 and two signal sensors for left and right. If the robot is required to turn left, its left signal sensor is turned on whenever the vertical co-ordinate of its centre is within 25% and 75% of the height of the vertical corridor. Similarly when it is required to turn right its right sensor is turned on. The robot needs to remember the signal it has received and act on it when it reaches the end of the corridor. When the robot reaches the point 90% of the vertical corridor's height from its base, it is re-placed at the same vertical position in the middle of the corridor (with a random offset of $[-1:1]$) at a random orientation within $[-0.2:0.2]$ radians of the vertically-upward direction. This prevents the robot from using its position or orientation as memory of the signal it received.

The robot has 150 time units in which to approach the goal which is in either the left or right corridor (depending which way the robot is required to go), in the middle, half of the corridor width from the end. Fitness is awarded for moving closer to the goal, as follows:

$$F = P\left(\frac{D-d}{D}\right) \tag{8.25}$$

where D is the robot's initial distance from the goal and d is its final distance from the goal (after 150 time units). P is a penalty term for hitting corridor walls, given by equation 8.18.

When the 150 time units are completed, the robot enters a new maze at the starting position. Each robot encounters 4 such mazes during its lifetime, in one of 4 possible sequences: LRLR, RLRL, LLRR or RRLL (where L indicates a maze with a left signal where the robot is required to turn left and R indicates a maze with a right signal where the robot is required to turn right). Its fitness for a lifetime is the mean of its fitness in each maze, and its overall fitness is the mean fitness over 4 lifetimes covering each possible sequence.

This task was evolved for growth and pruning 90 and 100, plastic and fixed controllers, as well as the special growth and pruning 95* controllers explained in section 8.2.12. The values of λ and μ (equations 8.11 and 8.12) were both set to 0.2. Each evolutionary run was allowed to proceed for 2000 generations, and 20 evolutionary runs were performed for every controller type.

This experiment is designed to shed light on open questions 1 and 4 of section 4.2.

8.2.8 Double T maze task

The double T maze is illustrated in figure 8.8. In this task the robot must respond to two different signals within a maze, taking two turns in the more complex doubly-divided maze based on the signals it receives to its left and right signal sensors.

At the beginning of the maze the goal is placed at one end of the horizontal corridor segment (vertically in the middle of the corridor, half of the corridor width from the end). The end it is placed at matches the signal that will be provided to the robot when it enters the 'signal active' zone between 25% and 75% of the vertical corridor height. When the robot reaches 90% of the corridor height it is replaced at the middle and rotated as described in section 8.2.7. It does not undergo such a movement at either end of the horizontal corridor.

When the robot comes within 30 units of the goal, it is moved to one of the four end points of the maze (half the corridor width from the end, in the centre horizontally). The end it is moved to matches the signal it receives in the 'signal active' zone of the horizontal corridor, so, for example if the robot receives a signal on its right sensor during its journey through the left portion of the horizontal corridor, it is required to turn right, so the goal is placed at the top of the left corridor segment. Similarly, if the robot receives a right signal when it is in the right section of the horizontal corridor, the goal is placed at the bottom end of the right-hand vertical corridor.

Fitness for the navigation of one maze is awarded as follows:

$$F = BO + P\left(\frac{D-d}{D}\right) \tag{8.26}$$

where D is the robot's initial distance from the goal and d is its final distance from the goal. BO is the bonus fitness it receives for hitting the first goal:

$$BO = \frac{250n}{T} \tag{8.27}$$

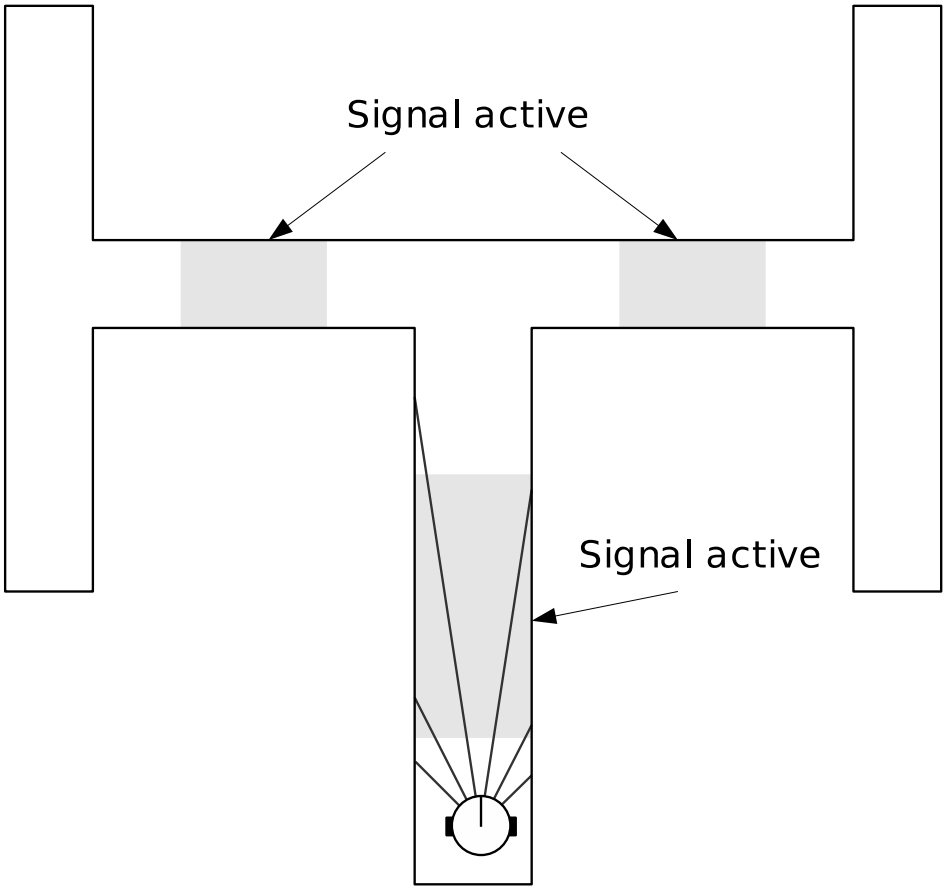

Figure 8.8: Double T maze task: the robot must navigate a maze taking two turns by following the signals applied to its left and right signal sensors. The signal in the horizontal corridor is only applied if the robot takes the correct turn in the vertical corridor.

where n is the number of times it reached the first goal (either zero or one) and $T = 250$ is the length of time allocated to each maze. P is the penalty for hitting walls as defined in equation 8.18.

The fitness of the robot over a lifetime is the mean of its fitness in each maze, and the final fitness is the mean of the lifetime fitnesses over four lifetimes each being one of the orderings (LL,LR,RR,RL), (LR,RR,LL,RL), (RR,LL,LR,RL) or (RL,LL,RR,LR) with LR meaning turn left then right and the other symbols having the corresponding meanings.

This task was evolved for growth and pruning 90 and 100, plastic and fixed controllers. The values of λ and μ (equations 8.11 and 8.12) were both set to 0.2. Each evolutionary run was allowed to proceed for 1000 generations, and 10 evolutionary runs were performed for every controller type.

This experiment is designed to shed light on open questions 1, 4 and 5 of section 4.2.

8.2.9 Learning task

The learning task takes place within the T maze scenario, but the robot is expected to navigate several mazes using only one signal. In the first maze it encounters, a signal is provided to specify whether it should turn left or right, but it then encounters 3 more mazes in succession in which it is required to turn in the same direction. No signal is provided in these last 3 mazes, and the robot is required to remember the original signal received throughout its lifetime. This task dramatically increases the length of time over which the robot's memory must operate.

The fitness of the robot in a maze is the same as in section 8.2.7 (equation 8.25), where the goal is placed in the same position in each maze encountered during a lifetime. The robot's fitness for a lifetime is found by taking the mean of its fitness in each of the 4 mazes it encounters, and its final fitness is taken to be the mean of its fitness over 4 lifetimes: 2 starting with a left signal and 2 with a right signal.

This task was evolved for growth and pruning 90 and 100, plastic and fixed controllers. Each evolutionary run was allowed to proceed for 2000 generations, and 20 evolutionary runs were performed for every controller type.

This experiment is designed to shed light on open questions 1, 4 and 5 of section 4.2.

8.2.10 Re-learning task

The re-learning task is identical to the learning task except that the robot is required to change its behaviour after having performed it several times. It is often found in learning situations that robots may become 'stuck' in a certain behaviour after performing some learning, making them incapable of adapting to new challenges. This task tests the ability of the developmental robots to reconfigure themselves to a new situation even after having learned another behaviour and become settled in it.

The lifetime of the robot in this task consists of eight mazes. In the first maze the robot receives a signal, which it must remember and apply for that maze and the following three. During the fifth maze the robot receives another signal (which may or may not be the same as the original one) and for this and the following three mazes it must remember and apply that signal.

The fitness of a robot for any given maze is the same as in equation 8.25, with the goal placed in the left or right-hand corridor depending on where the robot is required to go. The lifetime

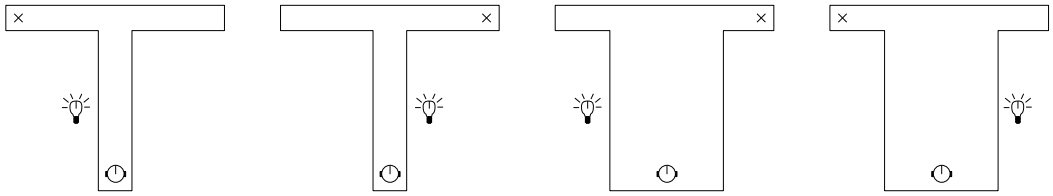

Figure 8.9: Flexibility task: the robot must respond to a difference in its environment (a wider vertical corridor) by changing its behaviour - in this environment the meaning of the signal is reversed. Here the light-bulb symbol represents the side on which the robot receives a signal, and the cross represents the location of the goal.

fitness of the robot is the mean over all eight mazes, and the final fitness of the robot is the mean over four lifetimes, covering the possible sequences LL, RR, LR, RL (where L indicates a maze with a left signal followed by three mazes with no signal where the robot is required to turn left, and R indicates a maze with a right signal followed by three mazes with no signal where the robot is required to turn right).

This task was evolved for growth and pruning 100, plastic and fixed controllers. The values of λ and μ (equations 8.11 and 8.12) were both set to 0.2. Each evolutionary run was allowed to proceed for 2000 generations, and 40 evolutionary runs were performed for every controller type.

This experiment is designed to shed light on open questions 1, 4 and 5 of section 4.2.

8.2.11 Flexibility task

One potential advantage of developmental agents discussed in section 4.1.3 is that they may be good at dealing with different initial conditions, and altering their behaviour to suit the environment within which they find themselves.

The flexibility task tests the ability of robots with the developmental controllers to perform different behaviours depending on the type of environment in which they are placed. This task takes place within the T maze context, where the robot is required to follow the signal under some circumstances and do the opposite of the signal in others. The different circumstances depend on the width of the vertical corridor of the maze.

If the robot finds itself in a vertical corridor whose width is within the normal $[25 : 35]$ range, it is awarded fitness exactly as in section 8.2.7, being required to turn left when a left signal is received, and right when a right signal is received. However, if it finds itself in a vertical corridor whose width is much wider (in the range $[95 : 105]$), it is awarded fitness for doing the opposite: turning right when a left signal is received, and left when a right signal is received. This situation is illustrated in figure 8.9.

During any lifetime, the robot only encounters either normal corridors, or wide corridors, but never both, since this task is designed to test the development of different behaviours when different environmental conditions prevail at birth.

The robot encounters four mazes during each lifetime, and its fitness for a lifetime is the mean

fitness over the four mazes. The robot's overall fitness is the mean of its fitness in each of the eight possible sequences of mazes, LRLR, RLRL, LLRR, RRLL, lrlr, rlrl, llrr, rrll where L indicates a normal corridor width with a left signal, R indicates a normal corridor width with a right signal, l indicates a wide corridor with a left signal and r indicates a wide corridor with a right signal. R and L have the goal placed on the same side as the signal and r and l have it placed on the opposite side.

This task was evolved for growth and pruning 90 and 100, plastic and fixed controllers. The values of λ and μ (equations 8.11 and 8.12) were both set to 0.2. Each evolutionary run was allowed to proceed for 2000 generations, and 20 evolutionary runs were performed for every controller type.

This experiment is designed to shed light on open questions 1, 4 and 6 of section 4.2.

8.2.12 Robustness to disruptions tests

For this section the performance of evolved robots in response to various disruptions is tested in order to understand whether the developmental controllers are robust to unexpected changes, as might be expected from the arguments outlined in section 8.1. Several disruptions are applied across a variety of robots evolved for two of the tasks described in previous sections, on robots with plastic and fixed controllers as well as growth and pruning 90 controllers and a modified growth and pruning controller, labelled 95*, which has a 0.95 probability of any synapse being alive at the beginning of the robot's lifetime, and an additional factor designed to improve robustness: in every second each live synapse has a probability of 0.001 of dying spontaneously, and each dead synapse has the same probability of re-growing spontaneously. This adaptation is designed to force the controller to be constantly re-adapting itself to maintain the correct behaviour, which may cause it to be in a constant state of dynamic change which might make it better at adapting successfully to unexpected disruptions.

No further evolution is involved: robots generated by evolution under normal circumstances are tested for their performance in altered scenarios.

The disruptions are tested on the best robots evolved in each evolutionary run described in sections 8.2.4 (phototaxis) and 8.2.7 (T maze). The tests involve testing each robot generated from each of 10 runs for each task over 100 lifetimes under the altered conditions and finding its fitness score. The following sections describe the disruptions which are applied.

This experiment is designed to shed light on open questions 1, 4, 7 and 8 of section 4.2.

Overactive left motor

Robots were tested with the 'overactive left motor' disruption which meant that their left motor applied more force for the same level of activation than it applied during evolution. The right motor was unaffected, so the motion of the robot was governed by the following adapted versions of the equations in section 8.2.1:

$$\dot{\theta} = \frac{r(XF_l - F_r)}{20} \tag{8.28}$$

$$\dot{s} = -0.9s + (XF_l + F_r) \tag{8.29}$$

This disruption tested the capacity of robots to adapt to a large alteration in their actuation which modified the character of their motion. It was applied from birth for the entire lifetime of the robot.

The severity of this disruption, denoted by X, was varied, being set at 2, 3 and 4 in different tests.

Altered angle between sensors

The 'altered angle between sensors' disruption involved changing the angles over which the robot's sensors were spread either making them closer together, or further apart, keeping their directions symmetrical.

In the unaltered phototaxis task the two sensors were spread over an angle of $\frac{3\pi}{4}$, and in the unaltered T maze task they were spread over $\frac{\pi}{2}$. In both cases tests were made at angles of 0.3π, 0.4π, 0.9π and π.

This disruption tested the capacity of robots to adapt to a change in their sensory environment which did not radically alter the character of the response to stimuli (for example, in the phototaxis task, if the goal was on the left of the robot, the left sensor was still more actively stimulated) but altered the actual levels of stimulation in a given lifetime significantly, and altered the relationships between the sensors, making their output more or less correlated depending on whether the angle has been increased or decreased. This disruption was applied from birth for the entire lifetime of the robot.

Sensors rotated relative to motors

The 'sensors rotated relative to motors' disruption caused the robot's sensors to be rotated relative to its direction of facing, while the angle between the sensors remained constant.

In the phototaxis experiments, the robot performed under normal circumstances for 300 units of time, and for the next 300 time units the sensors began to rotate until at time unit 600 they had rotated to their full extent. The robot then lived for another 600 units of time, and its fitness evaluation was based only on its behaviour during the final 300 time units of its life. In the T maze task the robot's sensors were rotated at the beginning of its lifetime.

The angles of rotation tested were 0.15π, 0.25π and 0.35π radians, where a value of zero denotes the sensors in their normal positions.

This disruption tested the ability of robots to adapt to potentially misleading information about the direction of objects in its environment, and in the case of the phototaxis task, it tested this in a situation where time was allowed for adaptation to occur.

Sensor inversion

'Sensor inversion' refers to the swapping of the robot's sensors, so that sensors which previously faced to the left of the robot were changed to face to its right at the same angle, and vice-versa.

This disruption, applied at the moment of birth, tested the ability of the robot to adapt to a radical shift in the relationships between its sensory experience, its motor output and its behaviour.

8.2.13 Robustness in larger fixed networks

As will be seen in section 9.1.9, the growth and pruning controllers performed well in some of the robustness tests. In order to investigate further the reasons for this success, a further 20 evolutionary runs were performed for the T maze task using each of the 'medium fixed' and 'large

fixed' controllers described in section 8.2.2. The large fixed controllers have a similar number of genetically-set parameters to the growth and pruning controllers, and the capacity for long time constants, giving them a similar temporal range. This allows for comparison of the methods of controller generation without bias being introduced by the introduction of different genotype lengths or timescales over which they may retain state.

8.2.14 Synapse growth verses death comparison

In order to investigate the relative influences of growth and death, the ability of evolution to produce fit robots for the phototaxis and T maze tasks under different levels of synaptic growth and death were performed. The parameters λ and μ from equations 8.11 and 8.12 were varied over different evolutionary runs, and the fitness of the best robot from each run was measured. These trials used growth and pruning 100 controllers.

For each type of controller, 20 evolutionary runs were performed for each of the 25 combinations of possible values of λ and μ of $0, 0.025, 0.05, 0.075$ and 0.1, making a total of 500 runs per controller, each one for 1000 generations. All members of the last generation of each run were evaluated accurately (by taking the mean fitness over 100 lifetimes) and the fittest individual from that generation was selected.

The final fitness score for each combination of λ and μ was calculated by taking the mean fitness of the fittest individuals in all 20 of the runs performed for that combination.

This experiment is designed to shed light on open question 9 of section 4.2.

The following chapter details the results of the above experiments and provides analysis and discussion of these results.

Chapter 9

Part 2 - Results, Analysis and Discussion

9.1 Results

9.1.1 Phototaxis task

All controller types evolved to perform the phototaxis task successfully.

Figure 9.1 shows a summary of the mean of the 20 fitnesses achieved for each controller type, evaluating the best robot from the last generation of each evolutionary run over 100 lifetimes. The detailed results are tabulated in table 9.1 and illustrated in figure 9.2. Histograms of these results are shown in figure 9.3.

The fixed controllers achieved the highest fitness in this task , with plastic controllers doing well and the two growth and pruning controllers performing less well. However the growth and pruning controllers do perform phototaxis, turning rapidly towards the goal and approaching it directly: the lower fitness score is due slower movement, especially near the beginning of the lifetime.

The Mann-Whitney test shows that there is significant evidence to suggest that the fixed controllers perform better than the plastic ($p = 1.7 \times 10^{-10}$), growth and pruning 95* ($p = 1.1 \times 10^{-7}$) and growth and pruning 50 ($p = 1.8 \times 10^{-4}$) controllers, and that the plastic controllers perform better than the growth and pruning 95* ($p = 1.5 \times 10^{-11}$) and growth and pruning 90 ($p = 3.9 \times 10^{-8}$) controllers. There is no evidence from this test to suggest a difference in performance between the two types of growth and pruning controller ($p = 0.54$).

9.1.2 Corridor following task

All controller types except growth and pruning 90 evolved to perform the corridor following task successfully in some cases, but some of the runs of each controller type failed to evolve. When an evolutionary run failed, the robots were often able to navigate the first two corridor sections, but could not reliably make the final right turn to move up the last vertical section (this outcome happened in each of the growth and pruning 90 runs). The successful robots move quickly along each corridor section and make each turn, avoiding contact with the walls.

Figure 9.4 shows a summary of the mean of the 10 fitnesses achieved for each controller type, evaluating the best robot from the last generation of each evolutionary run over 100 lifetimes. The

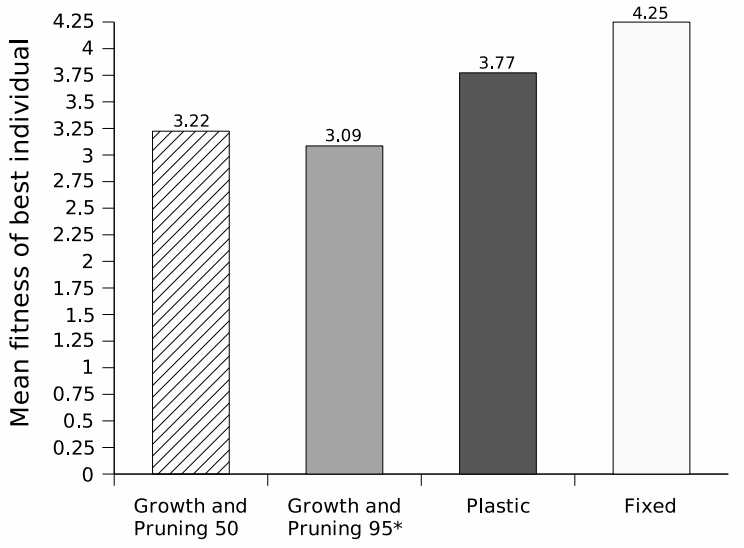

Figure 9.1: Phototaxis task: Each robot from the population at generation 1000 of each of the 20 evolutionary runs performed for growth and pruning 50, growth and pruning 95*, plastic and fixed controllers was evaluated over 100 lifetimes. The mean fitness of the best performing robot (over the 100 lifetimes) from each population was taken as the score for that evolutionary run, and the bars in the figure above show the mean score over 20 evolutionary runs for each type of controller.

Controller	**Mean**	**Fittest robot**				
Growth and Pruning 50	3.22	3.00	2.99	2.58	3.14	3.17
		3.53	3.73	2.71	3.25	2.64
		3.58	3.88	2.85	3.78	3.83
		2.35	3.69	2.95	3.09	3.75
Growth and Pruning 95*	3.09	3.22	2.22	3.31	3.19	3.01
		3.07	2.99	2.91	3.37	3.71
		3.34	2.63	2.98	3.02	3.36
		2.24	3.02	3.39	3.13	3.64
Plastic	3.77	3.93	3.89	3.62	3.69	3.60
		3.86	4.10	3.85	3.63	3.64
		3.67	3.60	3.77	3.87	3.74
		3.76	3.92	3.68	3.68	3.97
Fixed	4.25	4.32	4.35	4.21	4.45	4.50
		4.27	3.94	4.25	4.42	4.17
		4.11	4.43	4.26	4.00	4.12
		4.00	4.23	4.28	4.51	4.19

Table 9.1: Phototaxis task: The highest fitnesses achieved using different controller types. Each experiment was run 20 times, and the mean fitness over 100 lifetimes of the best individual from generation 1000 of each of the runs is shown in the right column, with the mean of these values shown to their left.

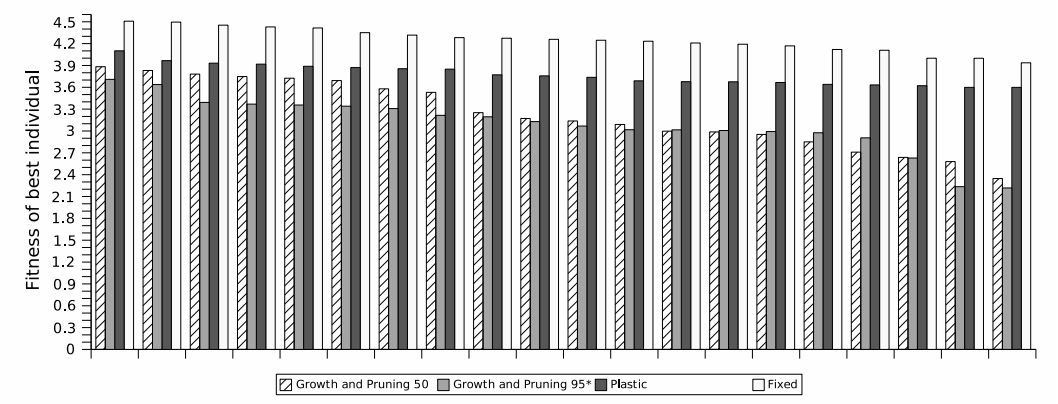

Figure 9.2: Phototaxis task: Fitnesses of the fittest robot from generation 1000 of each of the 20 evolutionary runs performed with each controller type. The scores are sorted in descending order to aid comparison. Exact values are shown in table 9.1.

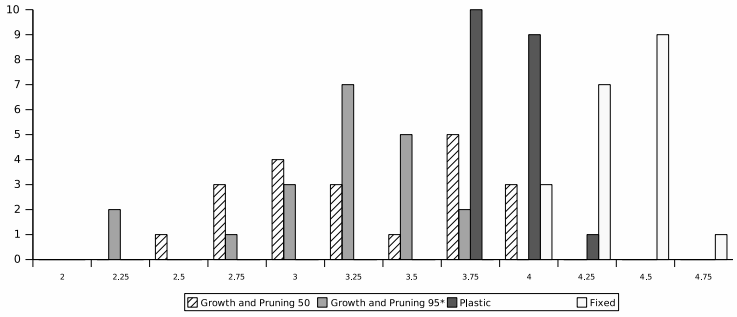

Figure 9.3: Phototaxis task: Histograms of the fitness scores shown in table 9.1. Where a column is labelled e.g. 0.6, it shows how many evolutionary runs resulted in robots whose mean fitness was $\in (0.55 : 0.6]$.

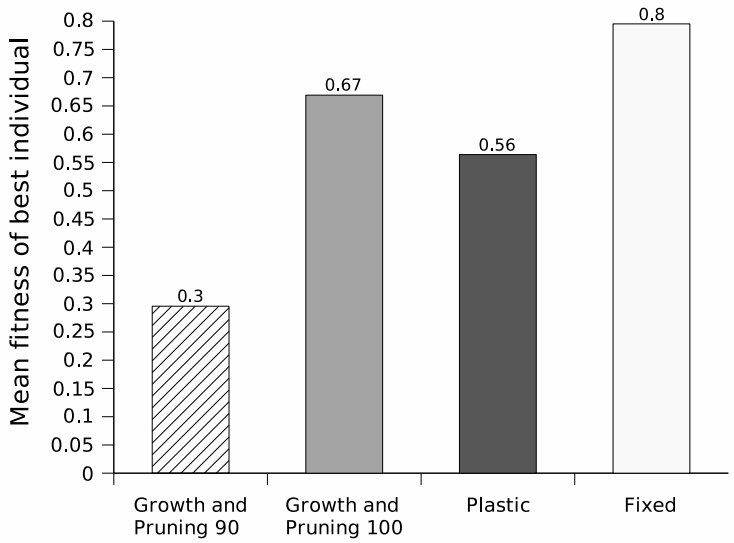

Figure 9.4: Corridor following task: Each robot from the population at generation 1000 of each of the 10 evolutionary runs performed for growth and pruning 90, growth and pruning 100, plastic and fixed controllers was evaluated over 100 lifetimes. The mean fitness of the best performing robot (over the 100 lifetimes) from each population was taken as the score for that evolutionary run, and the bars in the figure above show the mean score over 10 evolutionary runs for each type of controller.

detailed results are tabulated in table 9.2 and illustrated in figure 9.5. Histograms of these results are shown in figure 9.6.

The Mann-Whitney test shows that there is significant evidence to suggest that the fixed controllers perform better than the growth and pruning 90 ones ($p = 0.0028$). Due to the high variability of the results for other controller types, there is no evidence from this test to suggest differences between them (growth and pruning 90 vs. plastic $p = 0.20$, growth and pruning 90 vs. growth and pruning 100 $p = 0.061$, growth and pruning 100 vs. plastic $p = 0.80$, growth and pruning 100 vs. fixed $p = 0.39$, plastic vs. fixed $p = 0.17$).

9.1.3 Predictable change task

All controller types evolved successfully to perform the predictable change task, although the worst performers in terms of fitness were the growth and pruning controllers when put under additional selective pressure to undergo change at the time when the behaviour of the robot is required to change. The best performers were the plastic controllers, followed closely by the fixed and growth and pruning controllers.

Figure 9.7 shows a summary of the fitnesses achieved for each controller type. Growth and pruning with selection for change is marked as 'Growth and Pruning 100*'. The fitnesses shown are the mean bonus fitness received by the robot (*BO* in equation 8.22) over 100 lifetimes, since the final fitness scores are reduced in the situation where there is additional selection for change in the controller. Bonus fitness is an accurate measure of how well the robot does at this task, since it is proportional to the number of times it reached a goal during its lifetime. The detailed results are

Controller	**Mean**	**Fittest robot**				
Growth and Pruning 90	0.30	0.34	0.21	0.30	0.35	0.22
		0.26	0.30	0.30	0.34	0.32
Growth and Pruning 100	0.67	1.17	0.26	1.08	0.27	0.46
		0.51	1.11	1.10	0.39	0.33
Plastic	0.56	0.16	0.61	0.20	0.70	1.25
		0.63	0.79	0.55	0.30	0.45
Fixed	0.80	0.70	1.07	0.56	0.36	1.26
		0.35	1.30	1.29	0.74	0.32

Table 9.2: Corridor following task: The highest fitnesses achieved using different controller types. Each experiment was run 10 times, and the mean fitness over 100 lifetimes of the best individual from generation 1000 of each of the runs is shown in the right column, with the mean of these values shown to their left.

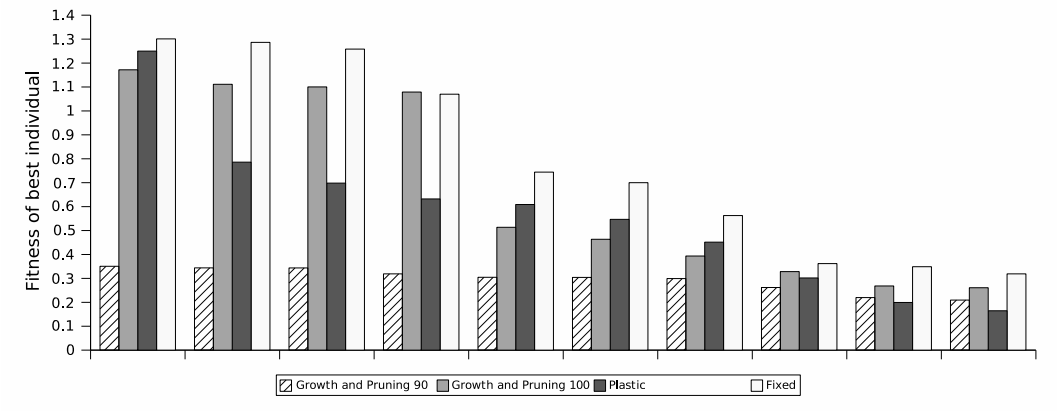

Figure 9.5: Corridor following task: Fitnesses of the fittest robot from generation 1000 of each of the 10 evolutionary runs performed with each controller type. The scores are sorted in descending order to aid comparison. Exact values are shown in table 9.2.

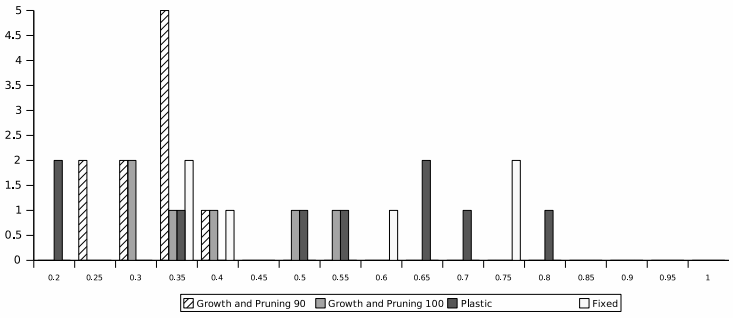

Figure 9.6: Corridor following task: Histograms of the fitness scores shown in table 9.2. Where a column is labelled e.g. 0.6, it shows how many evolutionary runs resulted in robots whose mean fitness was $\in (0.55 : 0.6]$.

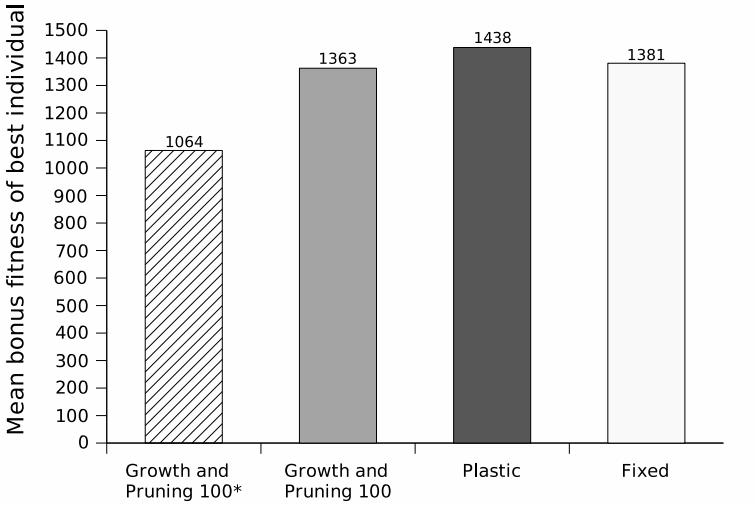

Figure 9.7: Predictable change task: Each robot from the population at generation 2000 of each of the 20 evolutionary runs performed for growth and pruning with selection for change ('Growth and Pruning 100*'), normal growth and pruning, plastic and fixed controllers was evaluated over 100 lifetimes. The mean bonus fitness of the best performing robot (over the 100 lifetimes) from each population was taken as the score for that evolutionary run, and the bars in the figure above show the mean score over 20 evolutionary runs for each type of controller.

tabulated in table 9.3 and illustrated in figure 9.8. Histograms of these results are shown in figure 9.9.

The behaviour of the robots was similar in each case, turning towards the top of the first corridor and navigating along the three corridors avoiding touch walls and turning quickly at each corner. When the end of the last corridor was reached the robots used a strategy of moving forward and rotating slowing until a goal was seen when they approached it quite directly, before beginning to rotate again to find the new goal when the goal was reached. Where performance was less good, the rotating strategy tended to result in an error where the robot moved around the outside of the maze, rather than approaching the goal.

Applying the Mann-Whitney test to examine the evidence for a difference between the growth and pruning controllers with and without selection for change in the controller gives $p = 0.02$. This provides evidence that the addition of this selection pressure has a detrimental effect on the expected final fitness. Applying the same test to compare the data for normal growth and pruning, plastic and fixed controllers provides no evidence to suggest that their median performances are different (growth and pruning vs. plastic, $p = 0.41$, growth and pruning vs. fixed, $p = 0.72$, plastic vs. fixed, $p = 0.44$).

9.1.4 T maze task

The T maze task consistently evolved very successfully for all the controller types except fixed. The fixed controllers were good at the task when they successfully evolved, but approximately half of the runs resulted in poorer controllers. The mean fitnesses are illustrated in figure 9.10, while the results of individual runs are shown in table 9.4 and illustrated in figure 9.11. Histograms of

Controller	**Mean**	**Fittest robot**				
Growth and Pruning 100*	1064	1405	492	1240	948	1478
		1290	1220	990	1938	1160
		1330	1302	1140	832	628
		1295	658	690	615	622
Growth and Pruning 100	1363	2108	1272	1732	1365	742
		1532	2125	1788	1318	1782
		1370	1108	1238	1182	915
		1128	702	1835	1235	778
Plastic	1438	1188	692	1410	1165	1472
		1410	1335	1848	1995	775
		750	1420	1920	1790	1855
		1508	758	2082	1778	1610
Fixed	1349	1658	1890	1310	1280	1658
		1460	1400	1390	970	1365
		1612	752	746	742	450
		1342	1802	1460	1512	2182

Table 9.3: Predictable change task: The highest bonus fitnesses achieved using different controller types. Each experiment was run 20 times, and the mean bonus fitness over 100 lifetimes of the best individual from generation 2000 of each of the runs is shown in the right column, with the mean of these values shown to their left. Here 'Growth and Pruning 100*' indicates robots under selection for change within the controller, and 'Growth and Pruning 100' indicates normal growth and pruning robots.

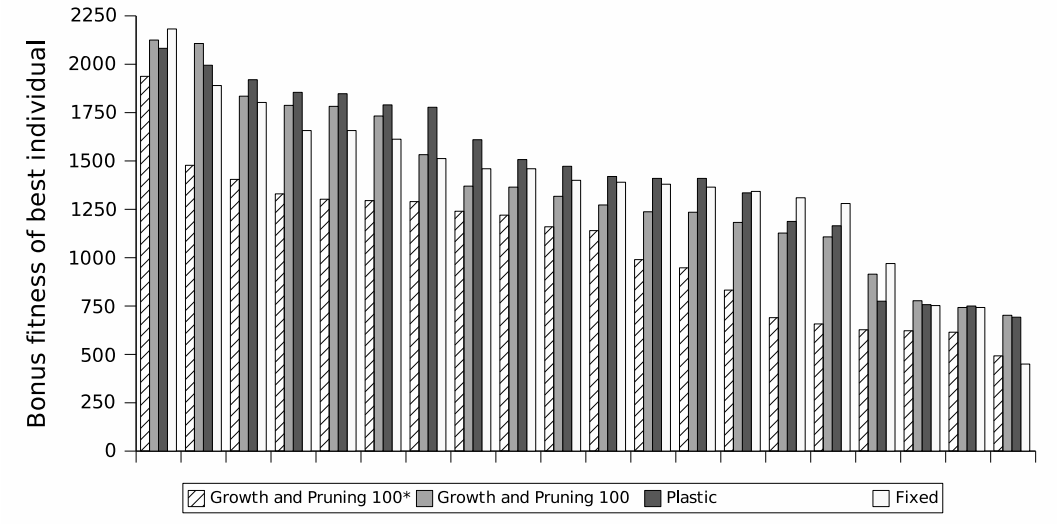

Figure 9.8: Predictable change task: Bonus fitnesses of the fittest robot from generation 2000 of each of the 20 evolutionary runs performed with each controller type. The scores are sorted in descending order to aid comparison. Exact values are shown in table 9.3.

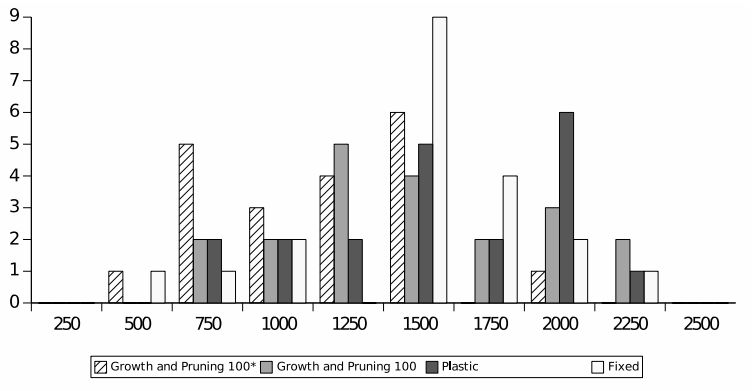

Figure 9.9: Predictable change task: Histograms of the fitness scores shown in table 9.3. Where a column is labelled e.g. 500, it shows how many evolutionary runs resulted in robots whose mean fitness was $\in (250 : 500]$.

Controller	**Mean**	**Fittest robot**				
Growth and Pruning 90	0.85	0.82	0.94	0.85	0.88	0.79
		0.84	0.81	0.88	0.86	0.84
Growth and Pruning 100	0.85	0.58	0.87	0.86	0.92	0.91
		0.91	0.95	0.90	0.74	0.91
Plastic	0.89	0.93	0.92	0.88	0.84	0.88
		0.95	0.86	0.91	0.90	0.87
Fixed	0.79	0.96	0.65	0.64	0.69	0.69
		0.90	0.69	0.90	0.89	0.93

Table 9.4: T maze task: Comparison of the highest fitnesses achieved in the T maze task using different controller types. Each experiment was run 10 times, and the mean fitness over 100 lifetimes of the best individual from generation 1000 of each of the runs is shown in the right column, with the mean of these values shown to their left.

the results are shown in figure 9.12.

The better fixed runs, and all of the runs for other controllers produced robots which consistently navigate the corridor quickly, without collisions with the walls and turn in the required direction. The poor fixed runs tend to react differently to different signals, usually failing to make a correct turn in one direction (sometimes simply slowing instead), yet succeeding in the other.

Applying the Mann-Whitney test to examine the evidence for a difference between the growth and pruning 90 controllers and plastic ones gives $p = 0.04$. This provides evidence that the plastic controllers may be expected to perform better than the growth and pruning 90 ones in this task. Applying the same test to compare the data for growth and pruning 100, plastic and fixed controllers provides no evidence to suggest that their median performances are different (growth and pruning 100 vs. plastic $p = 0.80$, growth and pruning 100 vs. fixed $p = 0.35$, plastic vs. fixed $p = 0.25$). This test also provides no evidence to suggest that the growth and pruning 90 controllers are better than the fixed ones ($p = 0.80$) or that growth and pruning 100 are better than

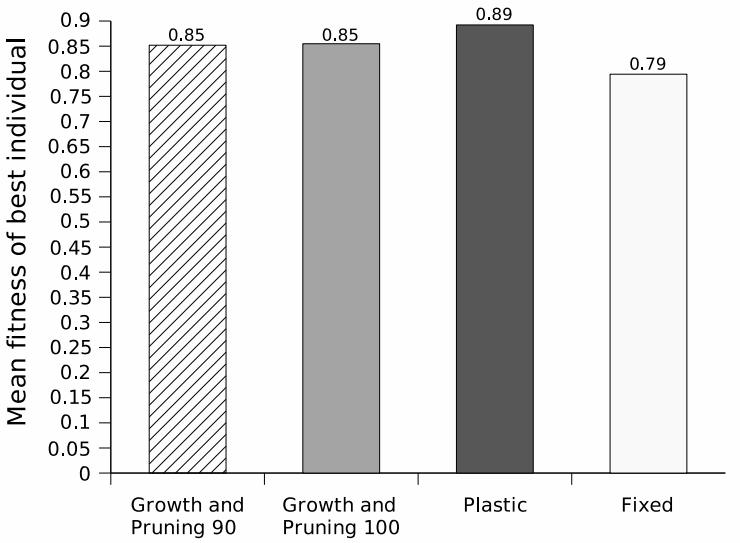

Figure 9.10: T maze task: Each robot from the population at generation 1000 of each of the 10 evolutionary runs performed for growth and pruning 90, growth and pruning 100, plastic and fixed controllers was evaluated over 100 lifetimes. The mean fitness of the best performing robot (over the 100 lifetimes) from each population was taken as the score for that evolutionary run, and the bars in the figure above show the mean score over 10 evolutionary runs for each type of controller.

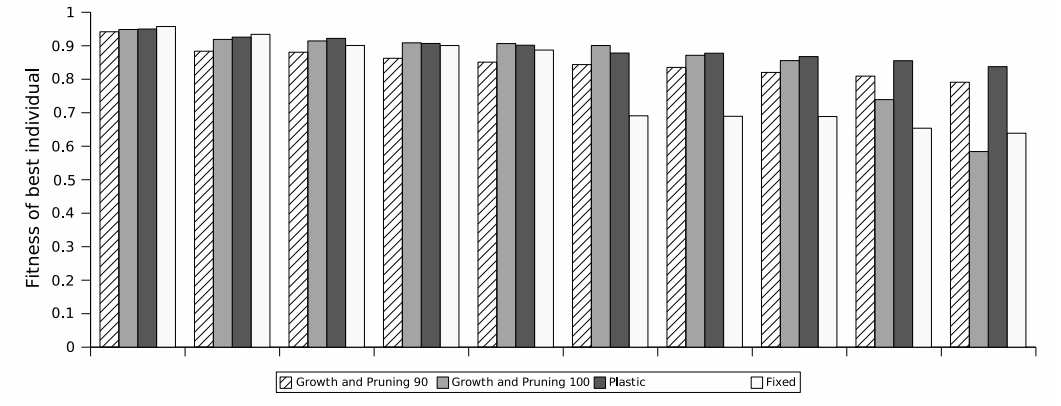

Figure 9.11: T maze task: Fitnesses of the fittest robot from generation 1000 of each of the 10 evolutionary runs performed with each controller type. The scores are sorted in descending order to aid comparison. Exact values are shown in table 9.4.

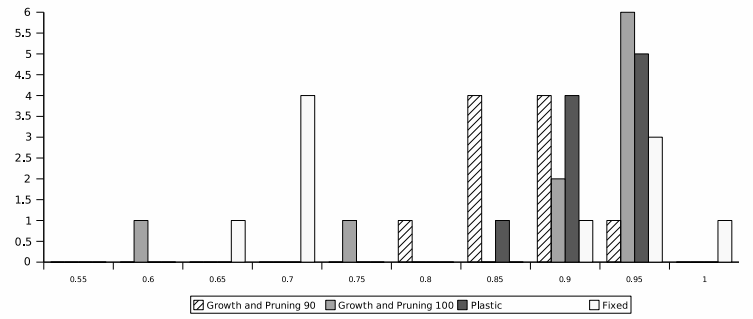

Figure 9.12: T maze task: Histograms of the fitness scores shown in table 9.4. Where a column is labelled e.g. 0.6, it shows how many evolutionary runs resulted in robots whose mean fitness was $\in (0.55 : 0.6]$.

growth and pruning 90 ($p = 0.19$).

9.1.5 Double T maze task

The double T maze task evolved very successfully for the plastic controllers and growth and pruning 100, and was less successful for the growth and pruning 90 controllers. The fixed controllers were good at the task when they successfully evolved, but several of the runs resulted in poor controllers. The mean fitnesses are illustrated in figure 9.13, while the results of individual runs are shown in table 9.5 and illustrated in figure 9.14. Histograms of the results are shown in figure 9.15.

The better fixed runs, and all of the plastic and growth and pruning 100 runs produced robots which consistently navigate the maze quickly, without collisions with the walls and turn in the required direction at both junctions. The poor fixed runs respond correctly to some signals and appear to ignore others. The growth and pruning 90 controllers behave erratically, performing the task well in most lifetimes, but occasionally becoming stuck always turning the same way, or not moving quickly enough along the corridor to reach the end.

The Mann-Whitney test shows that there is strong evidence for a difference in performance between the growth and pruning 90 controllers and plastic ones ($p = 0.0015$), between the two different growth and pruning controllers ($p = 7.6 \times 10^{-5}$) but no evidence for a difference between growth and pruning 100 and plastic ($p = 0.85$). Applying the same test provides no evidence for differences between growth and pruning 90 and fixed controllers ($p = 0.28$), between plastic and fixed controllers ($p = 0.35$) or between growth and pruning 100 and fixed ($p = 0.22$).

9.1.6 Learning task

In this task the plastic and growth and pruning 100 controllers outperformed the growth and pruning 90 and fixed controllers by a large margin. Growth and pruning 90 controllers performed particularly badly: the low scores were a combination of inconsistent, erratic behaviour, slow movement and, in some cases, not reacting to the signal in any recognisable way. Some mid and low-performing fixed controllers appeared to ignore the signal completely, turning left or right at random. Most of the plastic and growth and pruning 100 controllers performed the task con-

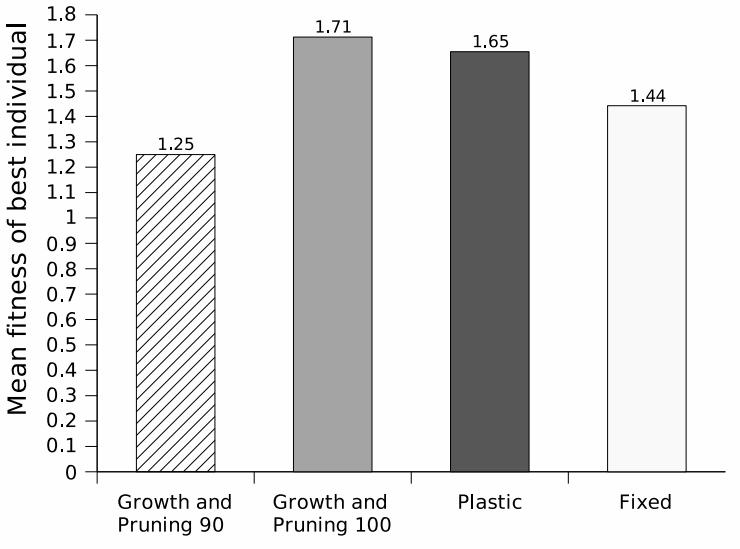

Figure 9.13: Double T maze task: Each robot from the population at generation 1000 of each of the 10 evolutionary runs performed for growth and pruning 90 growth and pruning 100, plastic and fixed controllers was evaluated over 100 lifetimes. The mean fitness of the best performing robot (over the 100 lifetimes) from each population was taken as the score for that evolutionary run, and the bars in the figure above show the mean score over 10 evolutionary runs for each type of controller.

Controller	**Mean**	**Fittest robot**				
Growth and Pruning 90	1.25	1.37	1.23	1.44	1.19	1.37
		1.27	1.14	1.15	1.07	1.27
Growth and Pruning 100	1.71	1.75	1.89	1.31	1.72	1.74
		1.77	1.77	1.73	1.74	1.70
Plastic	1.65	1.26	1.79	1.85	1.80	1.71
		1.70	1.69	1.77	1.75	1.24
Fixed	1.44	1.35	0.94	1.92	0.93	1.32
		1.11	1.53	1.58	1.83	1.90

Table 9.5: Double T maze task: Comparison of the highest fitnesses achieved in the double T maze task using different controller types. Each experiment was run 10 times, and the mean fitness over 100 lifetimes of the best individual from generation 1000 of each of the runs is shown in the right column, with the mean of these values shown to their left.

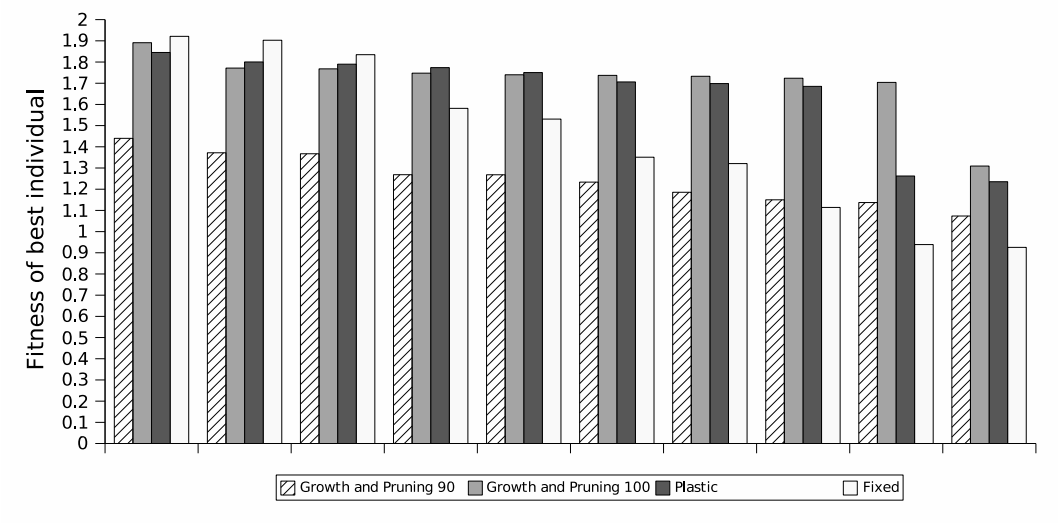

Figure 9.14: Double T maze task: Fitnesses of the fittest robot from generation 1000 of each of the 10 evolutionary runs performed with each controller type. The scores are sorted in descending order to aid comparison. Exact values are shown in table 9.5.

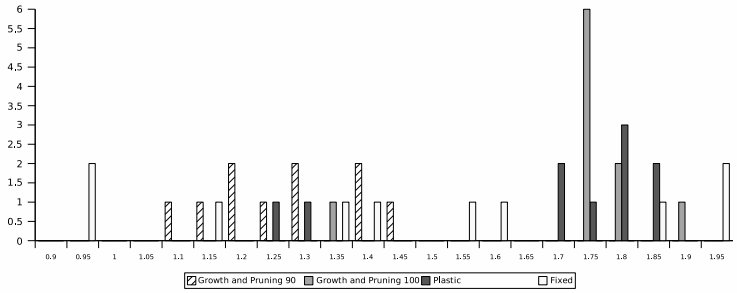

Figure 9.15: Double T maze task: Histograms of the fitness scores shown in table 9.5. Where a column is labelled e.g. 0.6, it shows how many evolutionary runs resulted in robots whose mean fitness was $\in (0.55 : 0.6]$.

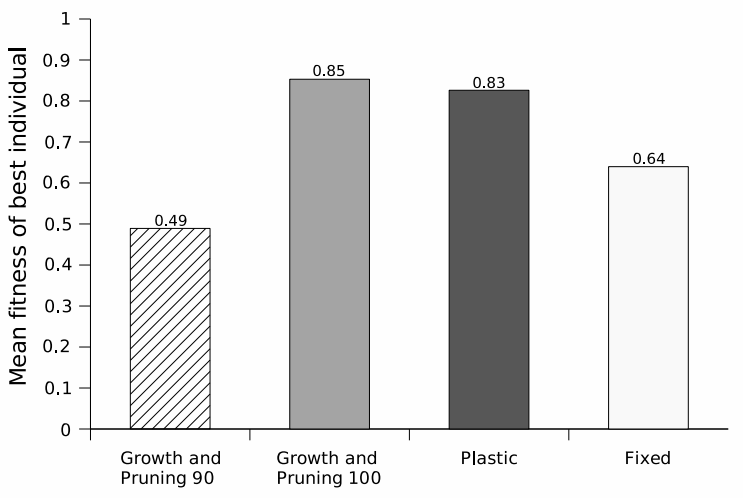

Figure 9.16: Learning task: Each robot from the population at generation 2000 of each of the 20 evolutionary runs performed for growth and pruning 90, growth and pruning 100, plastic and fixed controllers was evaluated over 100 lifetimes. The mean fitness of the best performing robot (over the 100 lifetimes) from each population was taken as the score for that evolutionary run, and the bars in the figure above show the mean score over 20 evolutionary runs for each type of controller.

sistently well, moving quickly up the vertical corridor and turning in the correct direction every time.

These results are summarised in figure 9.16, with details provided in table 9.6 and figure 9.17. A histogram showing the distributions of the results is shown in figure 9.18.

Testing for differences between each of the controller types using the Mann-Whitney test shows strong evidence that there is a difference in performance between the growth and pruning 100 and 90 variations ($p = 1.5 \times 10^{-11}$), the growth and pruning 100 and fixed controllers ($p = 2.1 \times 10^{-5}$), the plastic and growth and pruning 90 controllers ($p = 9.9 \times 10^{-9}$), the plastic and fixed controllers ($p = 3.7 \times 10^{-4}$) and the growth and pruning 90 and fixed controllers ($p = 5.4 \times 10^{-9}$). There is no evidence from this test for a difference between the growth and pruning 100 and plastic controllers ($p = 0.62$).

9.1.7 Re-learning task

The required behaviour evolved very successfully for this task using both the growth and pruning and plastic controllers. The fixed controllers were less consistently successful. Figure 9.19 gives an indication of the fitness of the best individuals that evolved using the different types of controller, showing the mean of the fitnesses of the best controllers over 40 runs for each controller type.

Table 9.7 shows the results in more detail, and figure 9.20 shows the values, sorted in descending order, from that table in graphical form. The plastic and growth and pruning controllers perform better than the fixed controllers, and at a similar level to each other.

The Mann-Whitney test shows that there is strong evidence that the plastic and growth and pruning controllers perform better than the fixed ones on this task (for growth and pruning vs.

Controller	**Mean**	**Fittest robot**				
Growth and Pruning 90	0.49	0.55	0.45	0.47	0.47	0.52
		0.50	0.46	0.49	0.49	0.54
		0.49	0.51	0.51	0.54	0.53
		0.47	0.47	0.45	0.44	0.46
Growth and Pruning 100	0.85	0.81	0.85	0.70	0.84	0.86
		0.93	0.96	0.78	0.83	0.82
		0.81	0.88	0.87	0.86	0.89
		0.89	0.80	0.86	0.93	0.91
Plastic	0.83	0.90	0.85	0.89	0.87	0.86
		0.75	0.88	0.88	0.46	0.85
		0.80	0.79	0.78	0.88	0.82
		0.86	0.92	0.88	0.86	0.73
Fixed	0.64	0.55	0.81	0.59	0.59	0.62
		0.59	0.61	0.59	0.57	0.51
		0.60	0.94	0.91	0.58	0.86
		0.61	0.63	0.57	0.54	0.53

Table 9.6: Learning task: The highest fitnesses achieved using different controller types. Each experiment was run 20 times, and the mean fitness over 100 lifetimes of the best individual from generation 2000 of each of the runs is shown in the right column, with the mean of these values shown to their left.

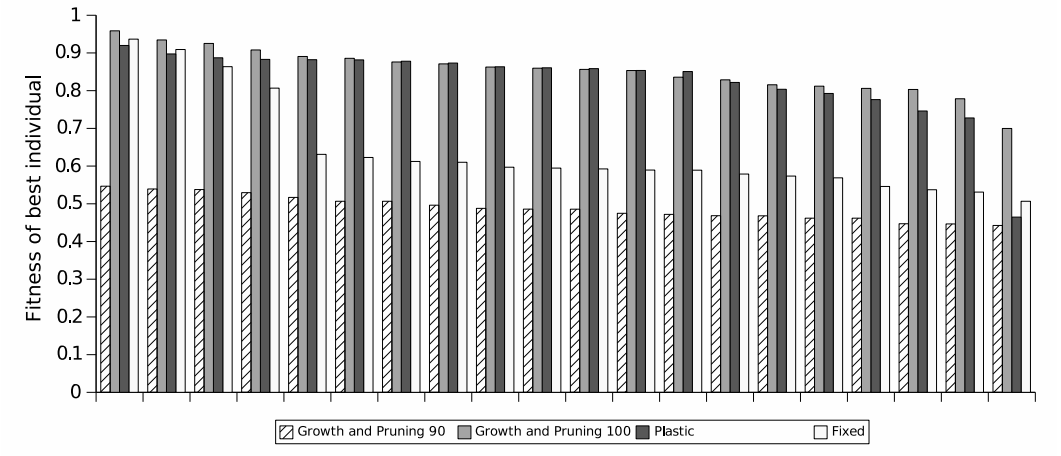

Figure 9.17: Learning task: Fitnesses of the fittest robot from generation 2000 of each of the 20 evolutionary runs performed with each controller type. The scores are sorted in descending order to aid comparison. Exact values are shown in table 9.6.

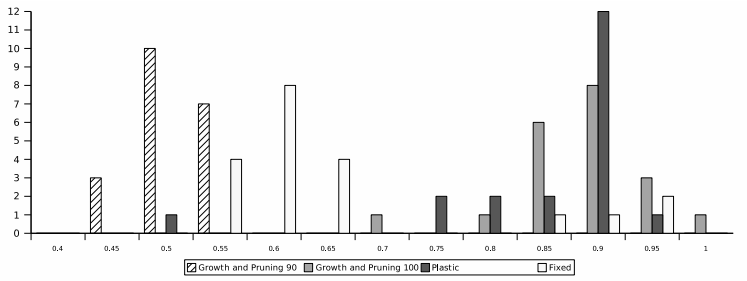

Figure 9.18: Learning task: Histograms of the fitness scores shown in table 9.6. Where a column is labelled e.g. 0.6, it shows how many evolutionary runs resulted in robots whose mean fitness was $\in (0.55 : 0.6]$.

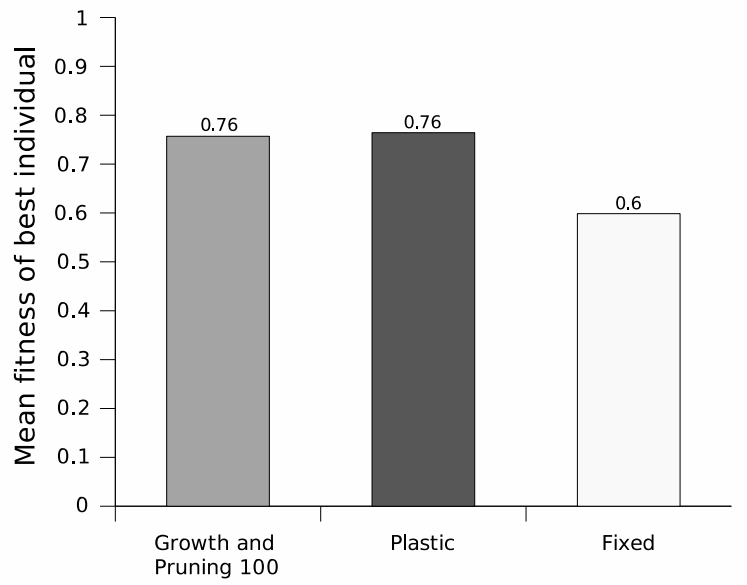

Figure 9.19: Re-learning task: Each robot from the population at generation 2000 of each of the 40 evolutionary runs performed for growth and pruning, plastic and fixed controllers was evaluated over 100 lifetimes. The mean fitness of the best performing robot (over the 100 lifetimes) from each population was taken as the score for that evolutionary run, and the bars in the figure above show the mean score over 40 evolutionary runs for each type of controller.

Controller	**Mean**	**Fittest robot**				
Growth and Pruning 100	0.76	0.87	0.85	0.78	0.82	0.57
		0.93	0.81	0.78	0.87	0.96
		0.83	0.91	0.93	0.88	0.83
		0.85	0.86	0.86	0.88	0.82
		0.59	0.73	0.66	0.69	0.76
		0.63	0.81	0.72	0.81	0.55
		0.58	0.90	0.82	0.55	0.61
		0.63	0.56	0.59	0.47	0.72
Plastic	0.76	0.85	0.75	0.87	0.81	0.87
		0.81	0.82	0.77	0.84	0.83
		0.61	0.73	0.93	0.71	0.90
		0.86	0.87	0.87	0.86	0.80
		0.77	0.85	0.72	0.36	0.86
		0.64	0.75	0.75	0.71	0.82
		0.80	0.77	0.50	0.66	0.51
		0.80	0.78	0.76	0.63	0.77
Fixed	0.60	0.87	0.59	0.59	0.55	0.88
		0.56	0.58	0.61	0.60	0.96
		0.61	0.57	0.70	0.61	0.58
		0.71	0.55	0.61	0.56	0.37
		0.64	0.50	0.57	0.56	0.61
		0.59	0.50	0.53	0.53	0.51
		0.52	0.53	0.59	0.61	0.62
		0.58	0.60	0.53	0.50	0.62

Table 9.7: Re-learning task: Comparison of the highest fitnesses achieved in the re-learning task using different controller types. Each experiment was run 40 times, and the mean fitness over 100 lifetimes of the best individual from generation 2000 of each of the runs is shown in the right column, with the mean of these values shown to their left.

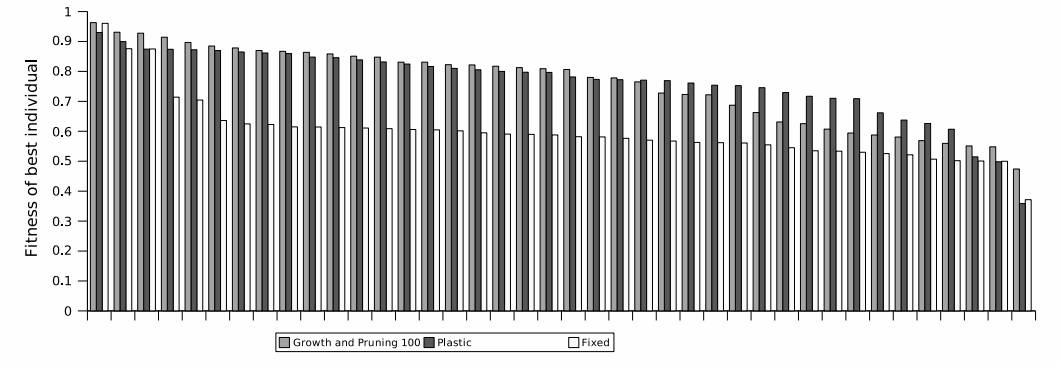

Figure 9.20: Re-learning task: Fitnesses of the fittest robot from generation 2000 of each of the 40 evolutionary runs performed with each controller type. The scores are sorted in descending order to aid comparison. Exact values are shown in table 9.7.

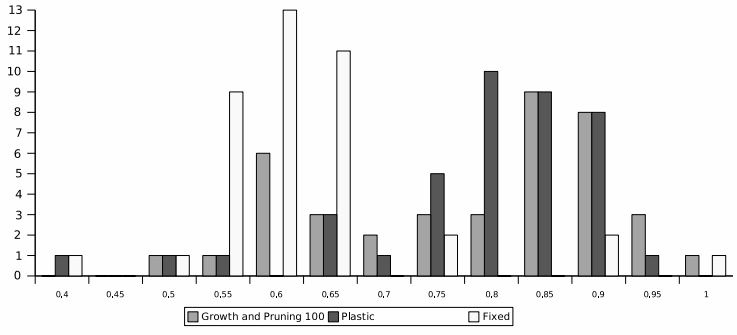

Figure 9.21: Re-learning task: Histograms of the fitness scores shown in table 9.7. Where a column is labelled e.g. 0.6, it shows how many evolutionary runs resulted in robots whose mean fitness was $\in (0.55 : 0.6]$.

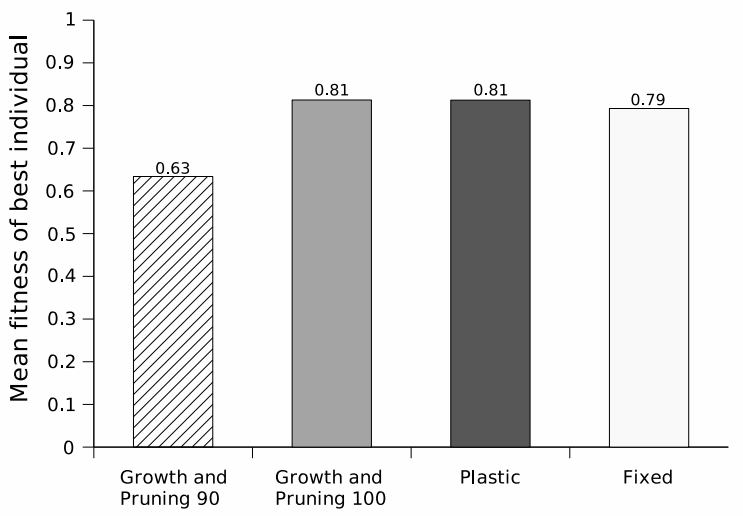

Figure 9.22: Flexibility task: Each robot from the population at generation 2000 of each of the 20 evolutionary runs performed for growth and pruning, plastic and fixed controllers was evaluated over 100 lifetimes. The mean fitness of the best performing robot (over the 100 lifetimes) from each population was taken as the score for that evolutionary run, and the bars in the figure above show the mean score over 20 evolutionary runs for each type of controller.

fixed $p = 8.4 \times 10^{-7}$ and for plastic vs. fixed $p = 1.0 \times 10^{-8}$). The same test does not provide any evidence that the growth and pruning controllers perform better than the plastic ones ($p = 0.98$).

Some of the robots described in this section are analysed in detail in section 9.2 in an attempt to understand how the required behaviour is generated by the different controller types and thus explain the differences in performance.

9.1.8 Flexibility task

The required behaviour evolved successfully for this task using the growth and pruning 100, plastic and fixed controllers. The growth and pruning 90 controllers were much less successful. Figure 9.22 gives an indication of the fitness of the best individuals that evolved using the different types of controller, showing the mean of the fitnesses of the best controllers over 20 runs for each controller type.

Table 9.8 shows the results in more detail, and figure 9.23 shows the values, sorted in descending order, from that table in graphical form. Histograms of the data are shown in figure 9.24.

The Mann-Whitney test shows that there is strong evidence for a difference in performance between the growth and pruning 90 controllers and all the other controllers ($p < 6.0 \times 10^{-4}$ in each case). The same test provides no evidence for any differences between the other controller types (growth and pruning 100 vs. plastic $p = 0.90$, growth and pruning vs. fixed $p = 0.65$, plastic vs. fixed $p = 0.68$).

9.1.9 Robustness to disruptions tests

In the phototaxis task with no disruption, the robots with fixed controllers performed best, followed by the plastic controllers, with the two types of growth and pruning following. When the

Controller	**Mean**	**Fittest robot**				
Growth and Pruning 90	0.63	0.67	0.60	0.65	0.60	0.66
		0.57	0.62	0.61	0.63	0.65
		0.67	0.83	0.59	0.64	0.70
		0.60	0.62	0.60	0.56	0.60
Growth and Pruning 100	0.81	0.90	0.77	0.83	0.80	0.70
		0.86	0.53	0.84	0.75	0.87
		0.92	0.70	0.85	0.86	0.88
		0.86	0.89	0.84	0.82	0.80
Plastic	0.81	0.82	0.86	0.84	0.91	0.76
		0.81	0.88	0.90	0.89	0.89
		0.80	0.77	0.84	0.54	0.67
		0.86	0.68	0.87	0.88	0.79
Fixed	0.79	0.93	0.60	0.68	0.94	0.87
		0.85	0.87	0.61	0.83	0.94
		0.91	0.91	0.57	0.93	0.90
		0.91	0.59	0.64	0.65	0.73

Table 9.8: Flexibility task: Comparison of the highest fitnesses achieved in the re-learning task using different controller types. Each experiment was run 20 times, and the mean fitness over 100 lifetimes of the best individual from generation 2000 of each of the runs is shown in the right column, with the mean of these values shown to their left.

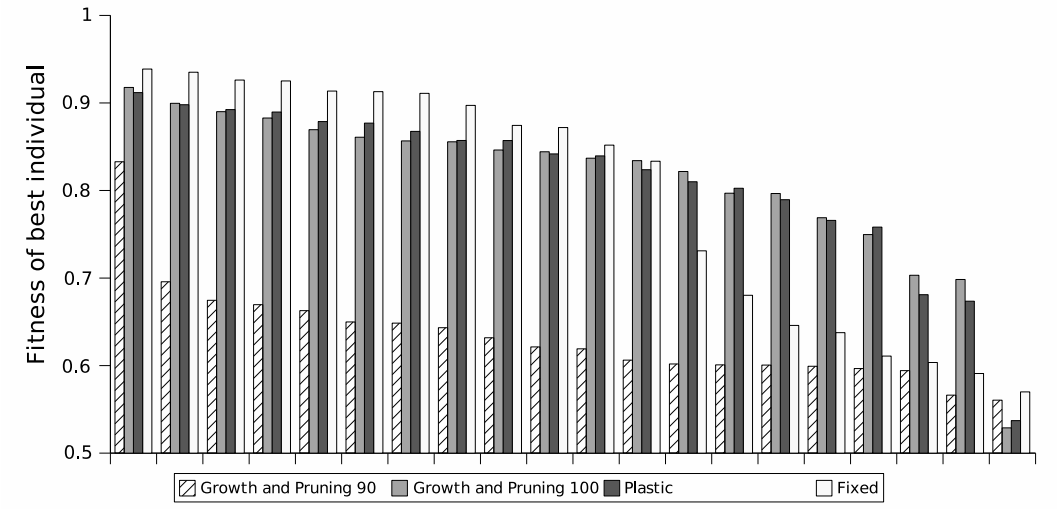

Figure 9.23: Flexibility task: Fitnesses of the fittest robot from generation 2000 of each of the 20 evolutionary runs performed with each controller type. The scores are sorted in descending order to aid comparison. Exact values are shown in table 9.8.

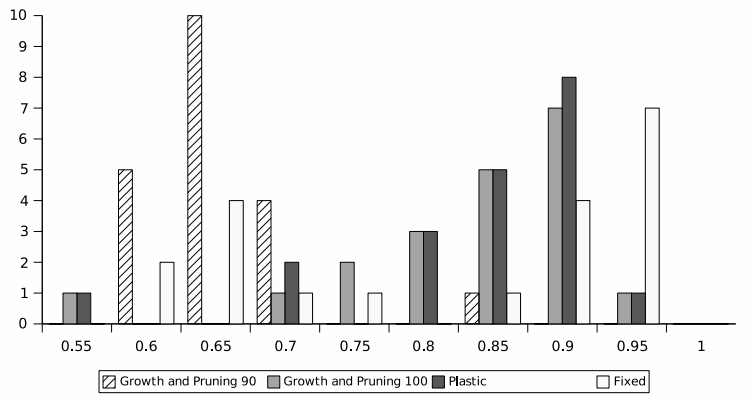

Figure 9.24: Flexibility task: Histograms of the fitness scores shown in table 9.8. Where a column is labelled e.g. 0.6, it shows how many evolutionary runs resulted in robots whose mean fitness was $\in (0.55 : 0.6]$.

disruptions were applied, as may be seen in figure 9.25, performance degraded quite uniformly, with no real advantage appearing to fall to the growth and pruning controllers over the others, and no advantage to plastic over fixed.

In the more complex T maze task with no disruptions the growth and pruning 95* controllers performed very badly, and the growth and pruning 90 controllers performed significantly less well than the plastic and fixed ones (growth and pruning 90 vs. plastic $p = 1.1 \times 10^{-5}$, growth and pruning 90 vs. fixed $p = 0.05$). The medium fixed controllers performed best, scoring significantly higher than growth and pruning 90 (growth and pruning 90 vs. medium fixed $p = 2.663e-07$), as did the large fixed controllers (growth and pruning 90 vs. large fixed $p = 1.3 \times 10^{-5}$).

However, when the disruptions were in force, the growth and pruning 90 controllers appeared to experience smaller degradations in performance than the plastic, fixed, medium fixed and large fixed ones. In the disruptions involving altered motor output, the growth and pruning 90 controllers moved from being significantly worse than fixed ones to significantly better (left motor times 2 $p = 0.015$, times 3 $p = 0.012$, times 4 $p = 0.0029$). They were also significantly better than medium fixed controllers (left motor times 2 $p = 1.3 \times 10^{-5}$, times 3 $p = 1.3 \times 10^{-5}$, times 4 $p = 2.0 \times 10^{-6}$) and large fixed controllers (left motor times 2 $p = 3.3 \times 10^{-5}$, times 3 $p = 3.0 \times 10^{-6}$, times 4 $p = 4.7^{-7}$).

As may be seen in figure 9.26, the degradation in performance of the growth and pruning 90 controllers was less than that of the plastic, fixed and large fixed controllers in most of the disruptions.

9.1.10 Synapse growth verses death comparison

The final fitness of each combination of λ and μ values (from equations 8.11 and 8.12) in the growth and pruning 50 controllers performing the phototaxis task is shown in figure 9.27. It may be observed that while the probability of death of synapses appears to have little effect on the fitness of the fittest individual, the probability of growth appears to affect the outcome quite considerably.

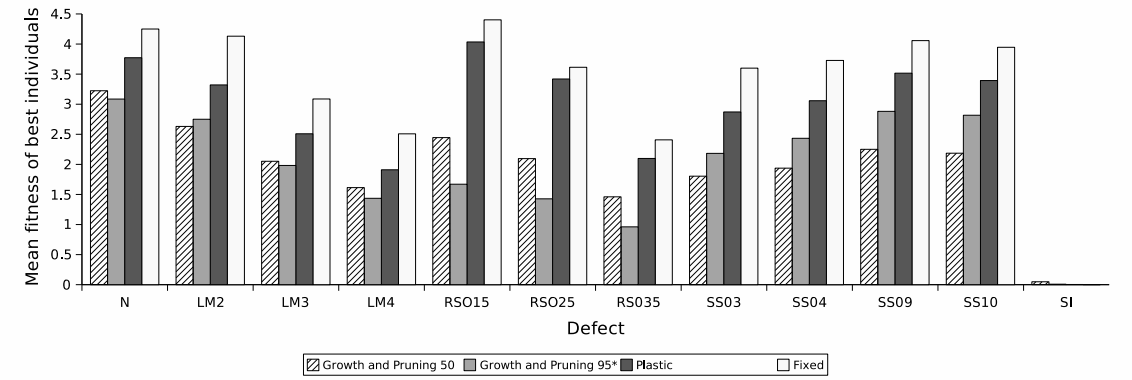

Figure 9.25: Phototaxis robustness: the mean fitness of robots performing phototaxis with different disruptions applied. The best robots from each of 10 evolutionary runs were chosen and evaluated without evolution with different disruptions applied. The mean fitness of the robot in each situation over 100 runs was found, and combined with the mean fitnesses for the other runs to provide an overall mean shown here. N denotes the performance of robots under normal conditions - as they were evolved. The LM2, LM3 and LM4 disruptions had the robot's left motor causing twice, three times or four times as much force as normal. The RS015, RS025 and RS035 disruptions had the robot's sensors rotated by 0.15π, 0.25π and 0.35π radians relative to normal. SS03, SS04, SS09 and SS10 refer to the sensors being spread over angles of 0.3π, 0.4π, 0.9π and π. SI refers to inversion of the sensors.

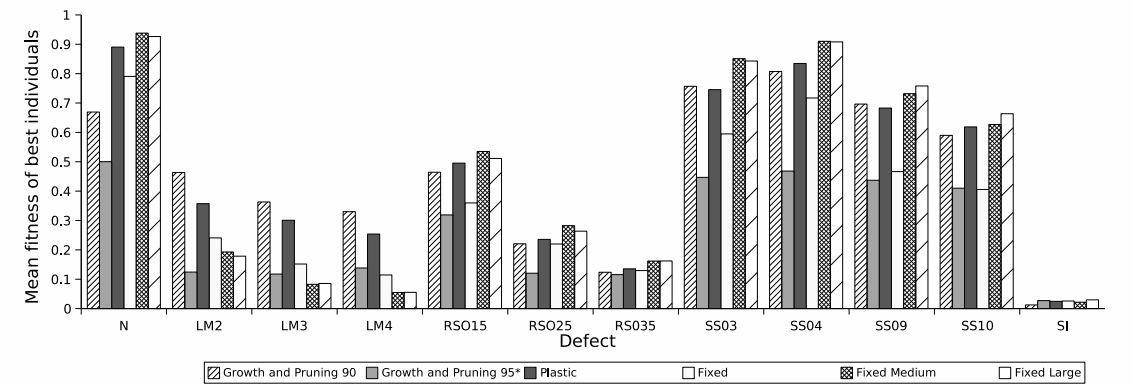

Figure 9.26: T maze robustness: the mean fitness of robots performing the T maze task with different disruptions applied. The best robots from each of 10 (20 for large and medium fixed) evolutionary runs were chosen and evaluated without evolution, with different disruptions applied. The mean fitness of the robot in each situation over 100 runs was found, and combined with the mean fitnesses for the other runs to provide an overall mean shown here. N denotes the performance of robots under normal conditions - as they were evolved. The LM2, LM3 and LM4 disruptions had the robot's left motor causing twice, three times or four times as much force as normal. The RS015, RS025 and RS035 disruptions had the robot's sensors rotated by 0.15π, 0.25π and 0.35π radians relative to normal. SS03, SS04, SS09 and SS10 refer to the sensors being spread over angles of 0.3π, 0.4π, 0.9π and π. SI refers to inversion of the sensors.

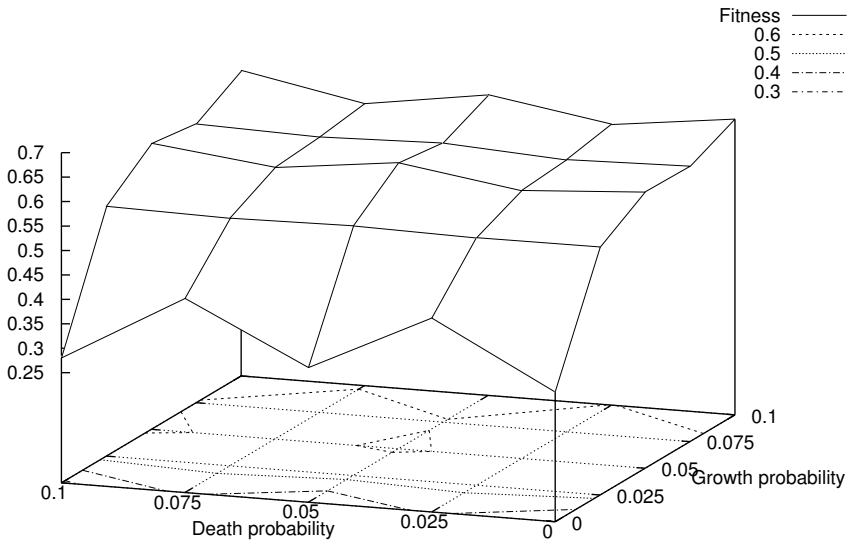

Figure 9.27: Mean fitness over 10 evolutionary runs of the fittest individual in the population after 1000 generations of the phototaxis task under differing levels of probability of growth (λ) and death (μ) of synapses when pre- and postsynaptic neuron firing rates were inside genetically-set ranges. Low likelihood of synapse death appears to have little effect on final fitness, whereas low likelihood of synapse growth has a negative effect.

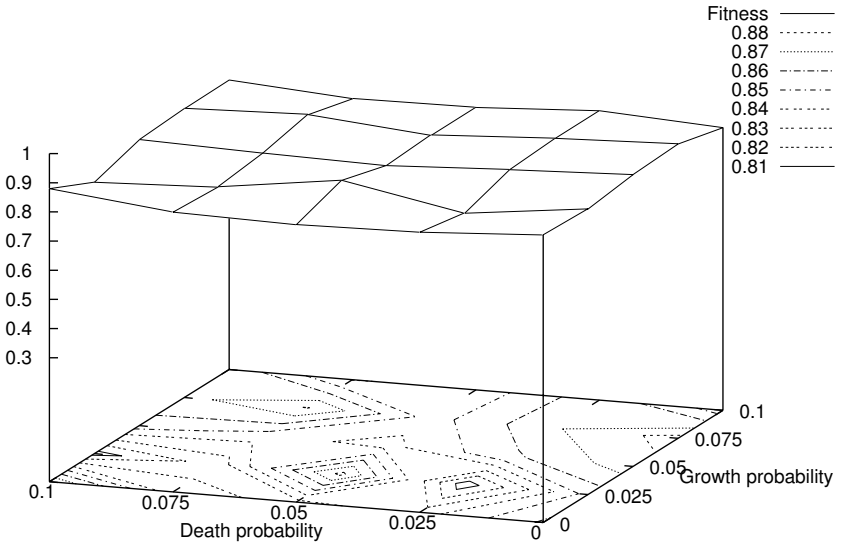

Figure 9.28: Mean fitness over 10 evolutionary runs of the fittest individual in the population after 1000 generations of the T maze task under differing levels of probability of growth (λ) and death (μ) of synapses when pre- and postsynaptic neuron firing rates were inside genetically-set ranges. Differing levels of synapse growth and death appear to have little effect on final fitness.

If synapse growth is quite unlikely or impossible, the fitness achieved is lower than if synapse growth is more likely. Observation of the evolved solutions suggests that the networks are converging onto fixed configurations of important weights, so any weights which are missing from that scheme must be grown at or near the beginning of the robot's lifetime.

In contrast, even if synapse death is impossible, good fitnesses may be achieved. This may be because synapse weight can still be suppressed to near zero, having the same effect as the death of that synapse, and because the effect of weights unneeded in the fixed configuration mentioned above may be small since their initial weights are random so they may often cancel out each other's effects.

The final fitness of each combination of λ and μ values (from equations 8.11 and 8.12) in the growth and pruning 100 controllers performing the T maze task is shown in figure 9.28. Here it appears that the probability of growth and death have little effect on final fitness.

In this task the robots possessed growth and pruning 100 controllers, all of whose synapses were alive at the beginning of the lifetime. This means that the effect found in the phototaxis task where low growth rates caused difficulty growing needed synapses was not found here, since all synapses were available unless they died during the robot's lifetime. It appears to offer no fitness advantage to be able to growth new synapses, or remove unneeded ones: this may be because all synapses are already available at the beginning of the lifetime, and because reducing the connection strength to near-zero has a similar effect to synapse death.

9.2 Analysis

In this section the behaviour, controller and developmental process of a robot with a growth and pruning 100 controller is examined in an attempt to shed light on the kinds of developmental process that may be generated by artificial evolution, and the kinds of dynamical system that may be constructed at the end of such a developmental process.

The robot under examination performs the re-learning task (see section 8.2.10 for a description of this task) with very high accuracy, achieving a mean fitness of 0.96 over 100 lifetimes. Its behaviour, controller and developmental pathways are examined in the following sections.

9.2.1 Behaviour

Observation of the robot, and the high fitness score it achieves, makes it clear that in terms of turning left or right at the appropriate times its behaviour is essentially perfect: over 100 lifetimes consisting of 25 of each of the scenarios LL, RR, LR, RL (section 8.2.10) it turns down the correct corridor segment in every maze it encounters.

The robot begins by orienting itself to face up the vertical corridor. Occasionally it does this by performing a full turn, but usually it turns immediately upward. It then moves quickly up the corridor, staying well away from the walls, near the centre. In mazes where it is required to turn left it tends to drift slightly to the right of the middle of the corridor, and in the opposite situation it drifts to the left.

When the robot reaches the T at the top of the vertical corridor it is moved to the centre and placed at a random orientation (facing generally upwards). It immediately turns in the direction of the goal and moves down the centre of the horizontal corridor in the required direction. When it encounters the end of the corridor it slows quickly to stop very close to the goal, usually reaching that point just as its time runs out.

During mazes where a signal is provided for the robot it slows slightly while the signal is active. When there is no signal present its progress is at a near-constant speed.

Several tests were performed to find out how the robot reacts to situations it did not encounter during evolution.

It is of interest to know whether the robot's 'memory' is trained to last for the time period used in evolution only (4 mazes) or whether it represents quite a permanent change in the dynamics of the controller. This was tested by presenting the robot with a series of mazes, starting with a left signal and then 19 mazes containing no signal. The robot continues to turn left in every maze, and when evaluated for 100 of these 20-maze lifetimes its mean fitness is 0.92, which is a very good score for this task. This suggests the robot's memory persists for much longer than was required during evolution.

In some cases when a behaviour is evolved its generation may be dependent on the specific details of the environment in which it is evolved. It is possible that the robot's ability to 're-learn,' modifying its behaviour when a different signal is received, is dependent on the timing of that signal, since this adaptation was only required during evolution after 4 mazes of one type had been experienced. A timing dependency might be during one maze, or over several of them.

The possibility of a dependency on timing during a single maze was tested by placing the robot into an environment with a longer vertical corridor - twice the length of those encountered during

evolution, in which the signal, when present, was only present in the lower section, for the same distance as in the original experiments. This meant the robot needed to navigate an extra length of vertical corridor, keeping its learning of the signal received intact, and turn at a later time in the correct direction. A longer time limit (300 simulated seconds instead of 150) was set to allow the robot time to reach the top of the corridor and make a turn. The robot is moderately successful in this scenario, normally making the correct turn, but occasionally making the wrong one, especially in the last of 4 mazes. It scores a mean fitness of 0.59 over 100 lifetimes.

Timing dependency over multiple mazes was tested using a lifetime consisting of a maze with a left signal, followed by one with a right signal, followed by another left and continuing alternating the signal and required direction every time, continuing for 20 mazes. The robot navigates this situation very successfully, receiving a mean fitness of 0.95 over 100 lifetimes.

The nature of the attractor states in the robot's behaviour might result in it becoming set into a particular pattern even when no specific trigger is received. This was studied by examining the behaviour when no signal at all is present. In this scenario the robot turns the first time at random, and from then often turns the same way for some or all of its lifetime. However, it is prone to change direction after several mazes in some cases. This implies that the initial random choice moves the robot into a controller-behaviour-environment attractor that has similar consequences to the dynamics caused by the receipt of a signal, but which is less stable, being disrupted in a few cases by the random elements of the simulation environment.

The overall picture of the robot's behaviour presented by these studies is of two very well-defined possible behavioural outcomes: the robot, even when it is performing badly, never stops, turns back or performs any behaviour except moving to the end of the vertical corridor and turning either left or right. These outcomes are stable to various different perturbations, most notably a longer vertical corridor section before the turn is made.

9.2.2 Controller

The behaviour described in the previous section is generated by a growth and pruning network consisting of inputs, output and neuron nodes joined by synapses whose weights may change and which may grow or die according to the interaction of genetically-set rules with node activations. Figure 9.29 shows a diagram of such a controller.

In order to understand how the behaviour is generated, it is useful to lesion elements of the controller and find how this affects the behaviour of the robot.

Table 9.9 shows the results of removing various combination of neurons from the controller. The neuron numbers are those shown in figure 9.29.

From the data in the table it seems reasonable to suggest that neurons 0 and 1 appear to have no impact on the robot's behaviour, while neurons 2 and 3 may be mainly involved with the learning behaviour (since their removal leaves wall-avoidance and movement intact), while neurons 4 and 5 may be responsible for wall-avoidance and movement. Of course, the interactions are likely to be much more complex than this in practice, but working within this framework may be helpful in understanding how the behaviour is generated.

The controller contains 96 synapses, so exhaustively disabling different combinations of them and testing their fitness would be very computationally intensive. An alternative approach is to

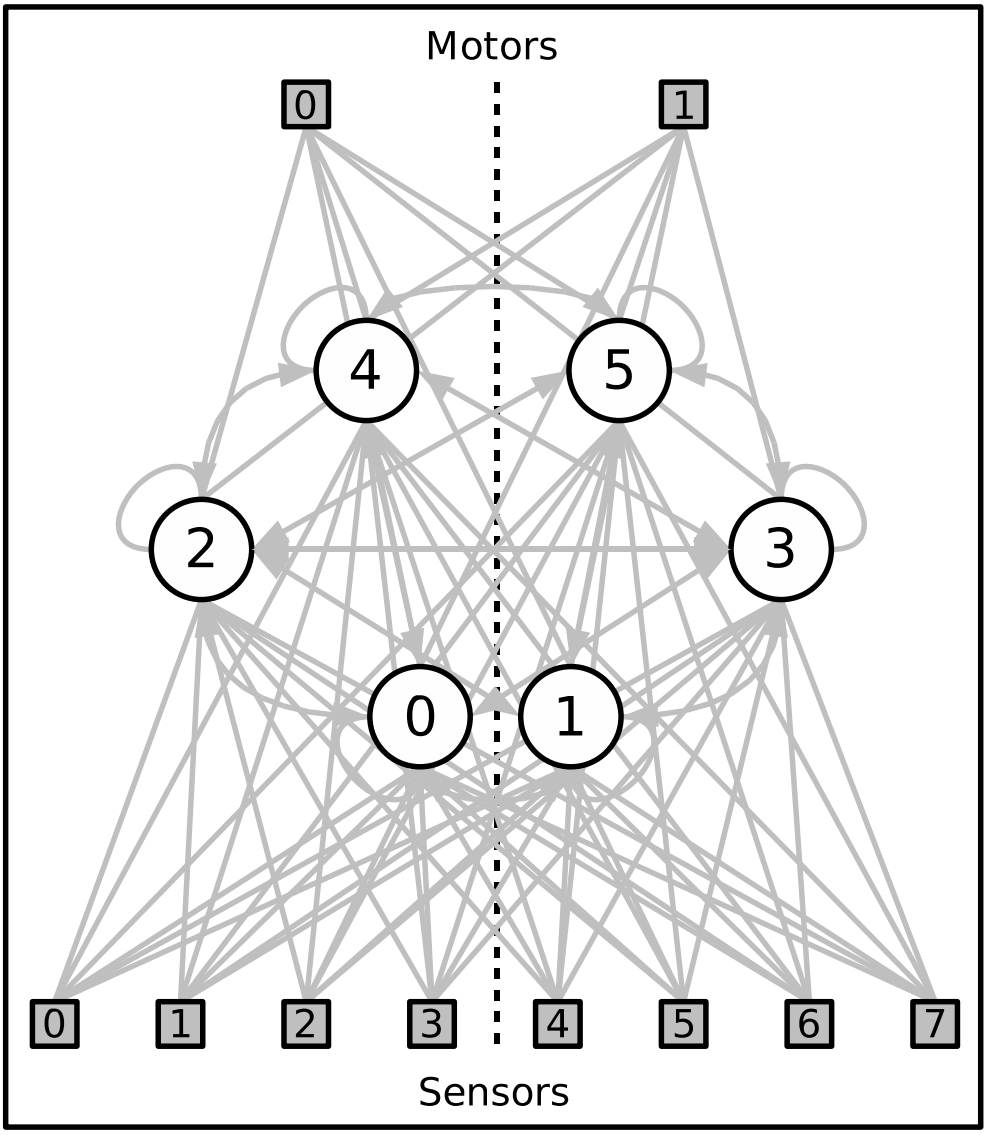

Figure 9.29: A growth and pruning controller. The sensors are fully connected to all of the neurons, which are fully interconnected, and fully connected to the outputs. Genetically-defined neuron and synapse properties are symmetrical about the dotted vertical line, but the developmental process and neuron activations are free to proceed asymmetrically.

Missing neurons(s)	**Fitness**	**Behaviour**
0 or 1	0.96	Indistinguishable from normal
2 or 3	0.50	Always turns one way (still moves and avoids walls)
4 or 5	0.51	Always turns one way, or occasionally simply circles.
0 and 1	0.95	Indistinguishable from normal
2 and 3	0.28	Moves upwards slowly, not appearing to respond to signal
4 and 5	0.08	No movement

Table 9.9: Lesioning neurons: the performance of the re-learning robot when one or two neurons are removed.

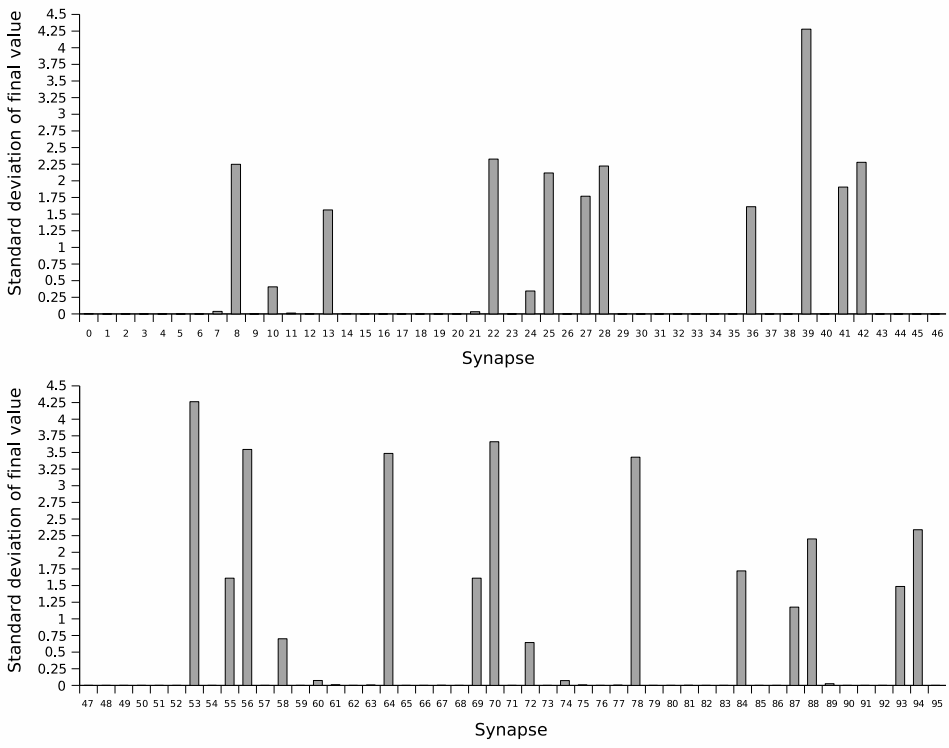

Figure 9.30: The standard deviations of the final synaptic weights of the robot's controller over 50 lifetimes containing different combinations of left and right signals.

attempt to reason about their roles by examining their behaviour during the robot's lifetime. A factor that might provide useful information about a synapse is whether or not it tends to settle to a steady value while the robot is behaving. By running the robot for many lifetimes and examining the synaptic strengths at the end of each one it may be possible to find which ones tend to enter steady states.

The robot was run for 50 lifetimes each consisting of different combinations of the possible signals LL, LR, RL and RR. At the end of each lifetime the synapse strengths were recorded, and finally a calculation of the standard deviation of each final synaptic strength value was found. A dead synapse was defined to have a strength of zero for the purposes of this statistic. Figure 9.30 shows the standard deviation of each synapse.

As may be seen from the figure, many of the weights have near-zero standard deviation, indicating that they normally settle to a particular value or die. The synapses with high variance consist of those which are essentially random, their strength not having a major effect on the behavioural outcome, and those which change according to the changing situations in which the robot finds itself.

If the robot is allowed to live for 100 standard lifetimes with all synapses with a standard deviation greater than 0.1 disabled, it scores an average fitness of 0.34, but on observation it appears to be performing wall avoidance and movement, albeit less well than the unaltered robot. It also appears to ignore the signal, suggesting that the signal reception and retention behaviour involves some of the variable synapses.

As a means of obtaining information on how the robot receives and remembers the signal, the

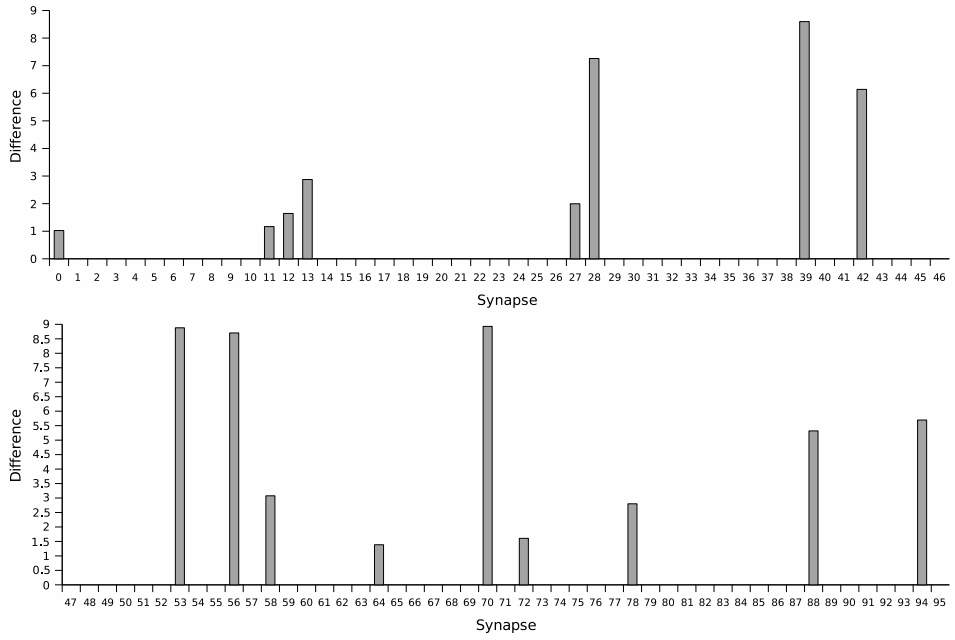

Figure 9.31: The difference in synaptic strengths during one lifetime after a left signal and a right signal.

state of the controller before and after the receipt of the second signal was recorded. The 'LR' scenario was used, where the robot receives a left signal in the first maze and must turn left for this and the next three, before receiving a right signal and being required to turn right for this and the last three mazes. The state of the robot's controller was recorded after 150 simulated seconds (at the end of the first maze, which contained the left signal), and 750 seconds (at the end of the fifth maze, which contained the right signal). Figure 9.31 shows the absolute differences between synapse strengths at these two moments.

Combining the previous hypothesis that the signal response and retention system relies on synapses that have a high variability with this new information about synapses that change in response to a change in signal, it may be that the crucial synapses are contained in the intersection of the sets of highly variable synapses and those that change in response to a change in the signal in the lifetime.

Taking these synapses, making the arrangement symmetrical and adding others as seemed appropriate, a process of trial and error was undertaken to find a reasonably minimal set of synapses that performed the task adequately. The final set chosen is illustrated in figure 9.32. It contains synapses that connect all neurons to the motors, the middle four sensors to neurons 4 and 5, and fully interconnects neurons 2, 3, 4 and 5.

The robot with only these synapses enabled received a mean fitness of 0.5 over 100 runs and when observed performs the require behaviour, albeit more slowly than the robot with a complete controller.

Examination of this robot performing the behaviour reveals how it works. When a signal is present, the synapse connecting the signal sensor to the opposite neuron (2 or 3) becomes negative, which, combined with the input from the signal sensor, reduces the activation of that neuron to

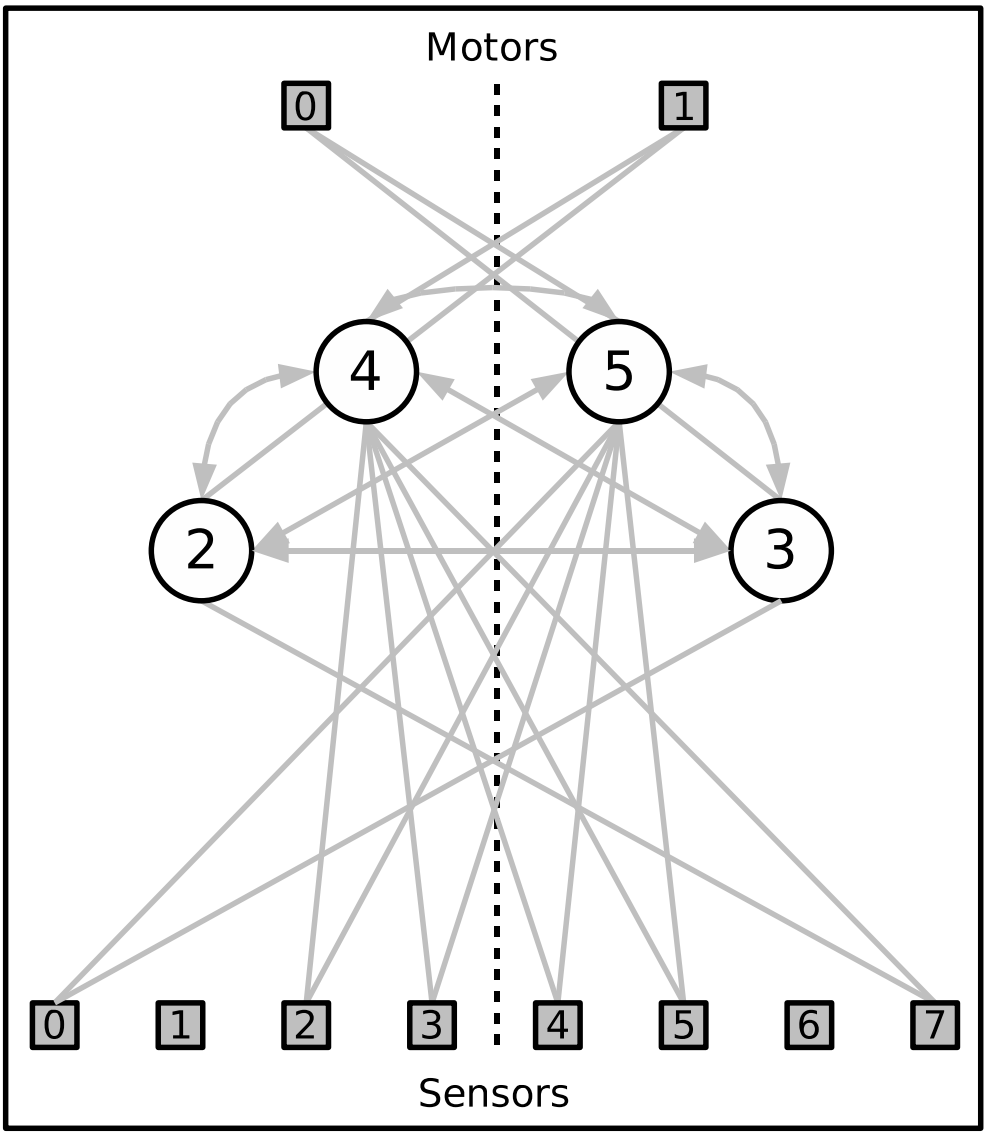

Figure 9.32: The controller used for the analysis of the re-learning task. Some of the synapses were disabled, leaving the ones shown above. The robot was still able to perform the task reasonably well using only these synapses.

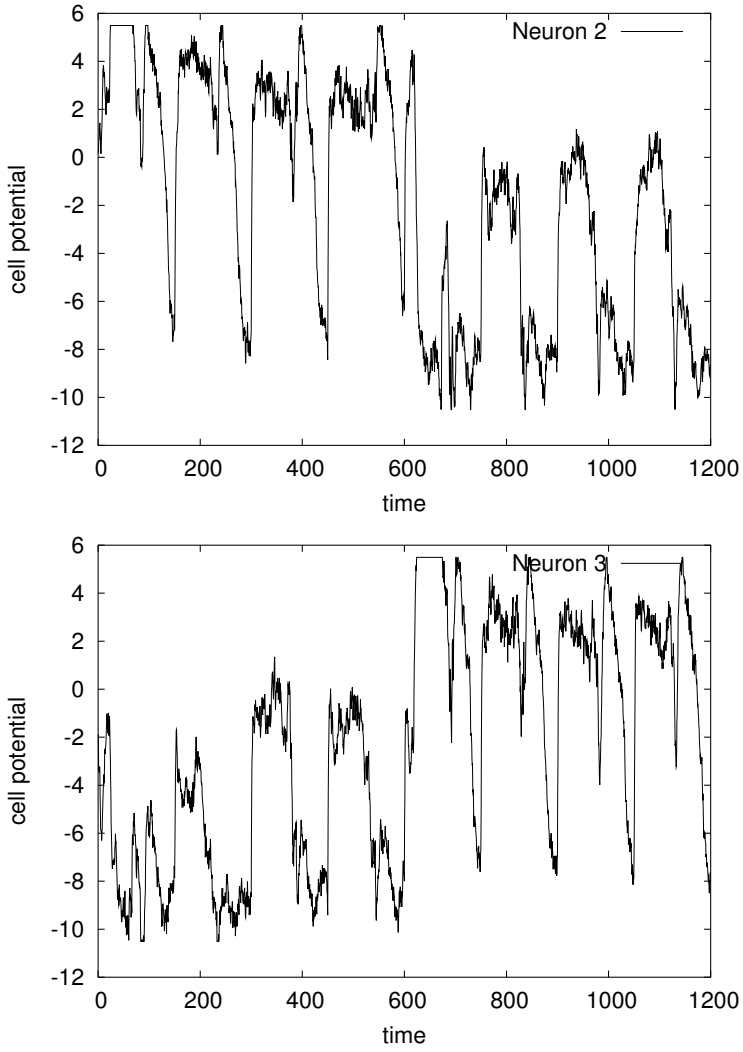

Figure 9.33: The cell potentials of neurons 2 and 3 throughout a lifetime consisting of 8 mazes. When neuron 2 is active, neuron 3 is suppressed, and vice-versa. The switch takes place when a signal is received telling the robot to start going the other way.

near its minimum value. This activity causes the synapse linking the neuron to the opposite motor to die, ensuring that neurons 2 and 3 only influence motor activity indirectly, through neurons 4 and 5. Once the controller is in this state, the low activation of this neuron persists, although it regresses towards a more moderate low value. This persistence is caused by negative reciprocal links between neurons 2 and 3, each of which has positive reciprocal links with the other neuron on its side (2 with 4 and 3 with 5). This means that when one of 2 and 3 becomes negative, the other becomes positive and the feedback loop with the other neuron on its side maintains this activation, suppressing the activity of the opposite pair.

Figure 9.33 shows the activity of neurons 2 and 3 over the course of a single lifetime consisting of one left signal maze followed by three no signal mazes, then one right signal and three no signal ones.

Neurons 4 and 5 have very low biases, and always have low activations, but strong positive

synaptic links to the opposite motors mean that they strongly influence the motor activations when they change state. Sensors are linked to both neurons 4 and 5, with negative links to the opposite neuron and positive links to the one on the same side. This combined with opposite links from 4 and 5 to the motors generates a simple wall-avoidance and movement system (corridor following) that is quite robust, and overrides any influence from the signals when sensor input is unbalanced.

When sensor input is balanced (as when the robot is at the decision point), the relatively slight difference in steady states of neurons 2 and 3 causes the robot to turn to the right when neuron 2 is low and to the left when neuron 3 is low. Once a small turn has been made the corridor following system completes the turn and continues the movement.

The key to the 'memory' of the robot is the dynamic balance between neurons 2 and 3 - when a signal is received it shifts one of these to become negatively activated, allowing the other to become positive, shifting into the alternative attractive state. This system is subservient to a simple corridor following strategy, only coming into play when the sensor activity is balanced.

The controller is produced through the developmental system, but does not use either synaptic plasticity or synapse growth and pruning to implement the memory system. The next section looks at how the behaviour described is produced through the developmental system.

9.2.3 Development

Almost all of the synapses mentioned in the above section have genetically defined death rules that are impossible to fulfill. This is achieved by setting the values from equation 8.12 to impossible values (i.e. $XD_1 > XD_2$ or $YD_1 > YD_2$). This allows the robot to rely on the presence of these synapses. A small number of other synapses not mentioned above are removed at the beginning of the lifetime. Removal of synapses provides a convenient way to exclude a synapse from disrupting a useful dynamical system.

A crucial synapse death occurs when the signal is received. When a right signal is received, the synapse connecting neuron 2 to motor 1 is killed due to the low activation of neuron 2. This occurs because this synapse has $XD1 = 0.03$, $XD2 = 0.57$, $YD1 = 0.31$, $YD2 = 0.71$, and here neuron 2 is X, and its activity only drops within the range $[0.03 : 0.57]$ when the signal suppresses it down to this level. Without removing this synapse, a direct link from neuron 2 to the motors is maintained, meaning that the memory system could influence the robot's movement directly, rather than being only an indirect influence through neuron 4. Similarly, when a left signal is received the opposite synapse is killed.

Most synaptic weights for the important synapses simply settle at ± 5, with the exception of those linking the signal sensors to neurons 2 and 3, which have an equation of change as follows:

$$\dot{w} = \alpha(-0.72x + 0.19y + -0.72xy) \tag{9.1}$$

This is equation 8.9 with the genetically-set values for this synapse shown. When x becomes positive due to the signal activation, and y is relatively near zero, this is negative, causing the synapse to become negative, and in turn reducing the activation of the neuron.

Thus most of the controller structure is encoded through the plasticity and growth and death processes in reasonably straightforward ways, but in a few places the relationship is a little more complex.

9.3 Discussion

All of the different controller types were able to evolve to perform the simpler tasks, although the growth and pruning controllers often performed worse than the plastic and fixed controllers, possibly due to the extra overhead of development being required, and the noise involved in beginning a lifetime with a random selection of synapses intact.

In the more complex tasks, the developmental controllers did better, outperforming the fixed controllers in several environments, and performing at least as well as the plastic controllers. However, the controllers which started with only 90% of their synapses alive showed no advantage in any of the tasks except when tested for robustness to disruptions.

It is clear that all of the tasks designed to utilise the extra developmental properties of the growth and pruning controllers were in fact possible to perform very well with the plastic controllers, and many of them were also possible with fixed controllers.

It has been shown that the explicit developmental process of the growth and pruning controllers tends to produce controllers which are significantly more robust than standard fixed controllers in the face of certain disruptions. This is true even when the comparison is made between controllers with similar numbers of free parameters under evolutionary control, and with similar ranges of temporal behaviour. Discussion of possible explanations for this outcome are discussed in the next chapter.

The idea that growth and pruning of synapses may be useful in tasks such as the predictable change and flexibility tasks appears to have lost credibility from these results, and the idea that selection for change in the controllers on the predictable change task would be useful appears to have been contradicted, as has the idea that the introduction of spontaneous growth and death of synapses would make the evolved robots more behaviourally robust.

In terms of the tests for how useful growth and death of synapses are relative to each other, only tentative conclusions may be drawn. In the experiment which used growth and pruning 50 controllers where the robot began its lifetime with many synapses missing, increasing the probability of growth was found to increase the mean fitness of robots evolved, but in the experiment with a more complex task but using growth and pruning 100 controllers, increasing growth probability appeared to have no effect on mean fitness. In both cases, increasing the probability of death of synapses appeared to have no effect.

Discussion of these and other issues may be found in the following chapter, along with some conclusions and suggestions for further avenues of research arising from both pieces of experimental work.

Chapter 10

Discussion

This thesis describes the motivations, methods and results of an attempt to design an evolvable developmental controller system useful for practical robot design and for the investigation of developmental phenomena.

Two different developmental controller systems were designed, evolved in different environments, and studied along with more commonly-used neural network systems, including plastic networks, which also explicitly implement developmental processes. The second developmental system was designed incorporating the lessons learned from working with the first.

The purposes of this work were to design tools useful in the study of developmental systems, and to learn about development. Section 10.1 describes the tools developed and their applicability, and section 10.2 outlines what has been learnt about each of the open questions explored.

Several issues of interest were raised by the problems encountered and the results found during the course of design and experimental work. Some of these are discussed further in sections 10.3 and 10.4.

Finally, section 10.5 looks at what conclusions may be drawn, and draws out the most important guidelines and advice for those considering continuing work in this area.

10.1 Development of tools

In the course of this work it has become clear that evolving developmental systems is not as straightforward as it might appear. In addition to the controllers developed (summarised in section 10.1.1), several techniques have been developed and lessons have been learned that may be of use to researchers intending to investigate further in this area. A selection of these techniques and methods is put forward in section 10.1.2.

10.1.1 Developmental Controllers

Chemical-guided growth networks

The controllers of part 1, chemical-guided growth networks, incorporate many ideas which may not have seen together before in evolutionary robotics studies of development.

The ideas of a single genotype for all units, spatial locations in controller space, chemical gradients affecting growth behaviour, influence of an agent's experience on its development and performance of behaviours of potential cognitive interest were combined in a single model.

The design is relatively simple to implement, and is readily modifiable to serve different experimental purposes, as exhibited by the variations described in terms of the addition of a 'clock' timer into the developmental process, and the use of different genotype mapping functions in different experiments.

Because the controllers, if the developmental process is frozen at any moment in their behaviour, behave exactly like the relatively well-understood CTRNN controllers frequently used in evolutionary robotics, they offer the chance to examine developmental phenomena in the context of a familiar system with dynamics that may be understood. (Note that the simulation time step in part one is adequate to simulate CTRNNs with a reasonable degree of accuracy, unlike the time step in part two.)

Thus it is hoped that variations on the theme of these controllers, possibly addressing some of the drawbacks found (e.g. the difficulty of building structure from a blank slate) may be useful in future studies of developmental systems within evolutionary robotics.

Synaptic growth and pruning networks

The controllers of part 2, being born partially out of the lessons learned in part 1, represent very practical tools for the study of developmental systems in many different contexts. They maintain many of the advantages of the controllers developed for part 1, crucially including the influence of experience on development and their basis in CTRNN networks, extended only where necessary to allow for developmental phenomena to be made explicit.

As may be seen by the wide variety of experiments which were performed, these controllers may successfully be evolved to generate many different behaviours. This rich variety of potential dynamics opens up the possibility of addressing many different questions in research into developmental systems. Since the controllers may be evolved to perform any of a large number of different tasks, it is likely that the behaviour required to investigate a phenomenon of particular interest may well be evolved in feasible time, allowing study of the way a developmental system can generate the behaviour of interest, and how it is affected by variations and changes of different characters.

The mechanism of synapse growth and pruning allows for many different experiments which might cast light on the equivalent phenomena in natural systems. The experiments investigating the relative importance of growth and death of synapses (section 8.2.14) are an example of the kinds of work which may be performed.

The flexibility of these controllers for performing varying experiments not only in terms of different environments and required behaviours but also in terms of the conditions and properties of the developmental system itself are demonstrated through the experiments into the influence of noise in the developmental system on robustness (section 8.2.12) and the usefulness of selection for particular developmental phenomena (in this case change at a specific time) in terms of producing particular behaviours (section 8.2.6).

10.1.2 How to evolve developmental systems

As outlined in section 4.1 there are many reasons why it might be useful to be able to evolve developmental systems: from expectations of the engineering uses for modularity and adaptability to pure scientific interest in the kinds of processes which produce useful systems. This section outlines some of the pitfalls and lessons learned during this thesis about how to evolve these systems in practice.

Recognise that it is not easy

There is a temptation when using evolution to design systems to assume that the evolutionary process is able to deal effectively with new complexity no matter where and how it is applied. In fact, certain types of structure and behaviour are much more easy to evolve than others. Developmental systems add a further level of abstraction between the behaviour being required and the genotype producing it, and this makes the task of designing a fitness landscape producing a smooth path between unfit and fit individuals much more complex. This extra complexity means that evolution may more easily become trapped in areas of low fitness unanticipated by the experimenter, and therefore a great deal more attention is needed to design an appropriate fitness landscape.

The problems encountered in the work described (and the solutions to some of those problems) have shown that the specific details of the developmental process have a radical effect on how easily a given behaviour may be produced in a robot using a genetic algorithm. Often the relevant details involve the fact that evolving developmental systems may settle on a behaviour that bypasses the capability for structural change, or uses that structural change directly to produce behaviour, instead of setting up a two-tier system with behaviour being generated by structure which is itself produced through development.

The pressure to act in a direct, one-tier manner may be quite strong in some cases. For example, a single evolutionary run of the developmental controllers from part one was performed for the multiple discrimination task, without the element of the fitness fitness function rewarding correlation between the robot's behaviour and required behaviour, over the course of 20,000 generations, and eventually a moderately high fitness score was achieved. When the evolved solution was examined, it became clear that small differences in the starting positions of circles and diamonds[1] were being used to discriminate, and the shape of each object was ignored. It turned out that the developmental system was growing neurons according to the positions of objects - a one-tier process - rather than growing a neural network which performed the discrimination. The evolved solution was extremely subtle (and highly sensitive to variation or noise) and certainly more complex than the evolved solutions found when the correlation element was added to the fitness function, and the difference between starting positions of objects was removed. This demonstrates the powerful effect of allowing evolution to start along a path through the fitness landscape which involves single-tier behaviour when two-tier behaviour is required.

One way in which the developmental process could have been prevented from taking this single-tier path might be to limit the developmental processes to work only over long timescales, preventing them from producing effective short-term behaviours. In this work, this was avoided since it was expected that interesting phenomena would be encountered through fast action of

[1] In this experiment circles were positioned at locations -50, -40, 30, ... whereas diamonds were positioned at locations -45, -35, -25, ... relative to the robot's starting position, as described in Beer (1996).

the development process, and these phenomena would be prevented from occurring if any 'speed limit' were imposed on the system. Examples of the kinds of phenomena expected include phase transitions in self-organising systems and rapid but long-lasting reactions to short-term stimuli. However, this kind of restriction may well be appropriate in studies not concerned with such high speed dynamics.

Thus, in order successfully to evolve developmental robots it must be recognised that simply inserting an arbitrary system of structural change is not enough: the system must be an appropriate one for the types of development and behaviour required, and the type of evolutionary process through which it is generated. It must also provide as direct a mapping as possible between the genotype and the quantity being selected for, either by decreasing indirection within the developmental process or by selecting directly for specific developmental outcomes. To achieve these aims the designer must make every effort to understand the problem domain before designing the system to fit not only the outcome but also the intermediate stages in both the evolutionary and developmental processes. Furthermore, he or she must anticipate the need to modify the system as designed to explore ways of allowing it to evolve in the directions intended.

The following sections outline some of the areas that should be given attention when designing developmental systems to evolve behaviours of non-trivial complexity.

Structure

A crucial element in the successful evolution of the developmental controllers studied in this thesis is the imposition of structure on the system. In some cases this involves fixing specific factors (such as the use of symmetry) and in other cases it involves supplying non-specific (random) structure which is used as raw material for the developmental process to work with.

The imposition of symmetry on the controllers of part one is required to allow the robots to evolve to perform the multiple discrimination task. Without symmetry in the developmental process, the task of developing a symmetrical controller (which is very useful in a symmetrical environment) is difficult, since it requires symmetry in the genetic values. If the robot mutates so that one half of its controller has a structure that could successfully perform the task, but the other half is different, its ability to perform the task may not be increased at all by the 'correct' half. Often each step in the evolutionary process must be stumbled upon simultaneously for both the left and right sides of the robot. This adds yet another degree of indirection between genotype and behaviour, and in the work performed for part one this indirection combined with the inherently indirect processes of development to prevent successful evolution within in a reasonable time without symmetry.

Of course, if for example the evolution of developmental symmetry is the matter under examination, it is important not to impose this constraint on the system being studied. However, here the evolution of a system with a fixed developmental process is being studied, and it makes sense to model that process as having symmetry where it may be useful for the task and environment being studied.

Another example of the advantages of imposing structure on the system is the fact that the layout of the neural networks in the growth and pruning networks is fixed, with each synapse able to grow or die, but without the need for neurons to 'find' each other in order to connect as in part one. This kind of situation is an example of how imposing structure can reduce the level of

indirection involved in the genotype-behaviour relationship.

Other examples of structures that might be useful to impose on controllers include forcing neural networks to be scale-free (since scale-free networks are found very commonly in nature and have been found to have very useful properties), and adding constraints to the weights of synapses entering a neuron to cause synapses to compete for the resources provided by neurons.

An important feature of the design of the controllers for part two is the imposition of random structure which may be manipulated by the developmental process. This structure allows development to channel existing activity towards a desired state, rather than generating entirely new structure through which activity may flow. This idea was introduced as an abstraction of the idea of the 'exuberant growth' of neural structure which may occur in animals before the structure which has grown is re-formed under environmental influence.

The success of the growth and pruning 100 controllers over the 90 variants demonstrates the importance to the success achieved in part two of this insight about providing structure which may then be manipulated during evolution.

It is important to note that the imposition of structure (especially specific structure like symmetry) normally represents a trade-off between the ability to evolve a solution and the addition of extra assumptions. Any structure that is forced upon the developmental process, or supplied to it when the robot is born, is an instantiation of a set of assumptions on the part of the experimenter. This trade-off is familiar to practitioners of genetic algorithm techniques since the evolution of complex outcomes often involves the imposition of specific constraints on the paths evolution may take. In all such cases, caution is advisable to ensure that the expected outcome is not being programmed in through the assumptions being made.

The explicit method through which the evolutionary process may be 'instructed' by the experimenter is the fitness function. The next section looks at how the evolutionary process may be influenced to follow useful paths using fitness, rather than structural constraints.

Fitness functions

The evolution of a discriminating robot using the chemical-guided networks demonstrates the importance of the fitness function in the process of evolving developmental systems. The fitness function must be used to shape the landscape navigated by evolution so that it leads towards good solutions. This can mean using powerful, highly specific fitness functions to ensure only 'correct' behaviour is rewarded (as in the case of the discriminating robot), and it can also involve the use of domain-specific knowledge to guide evolution down useful paths.

This domain-specific knowledge may involve understanding of the kinds of intermediate solution that will be useful stepping stones towards successful evolution (biasing the fitness function to encourage evolution towards these solutions) as well as knowledge of the kinds of structures that are expected in the controller, as was attempted in the predictable change task.

An interesting addition to this idea is that if a solution works in a certain way, it may be expected to fail under certain specific circumstances. If working in that way is required, or is assumed to be a useful stepping stone in the evolutionary process, it could be useful to reward the robot for failing under these specific circumstances. A reward for failure might seem counter-intuitive, but actually what is happening is a narrowing of the fitness function, strengthening its

ability to disassociate potentially good solutions from evolutionary *cul-de-sacs*[2].

The use of specialised fitness functions to evolve successful behaviours is, of course, simply another way of imposing structure on the system, and as such is subject to the same caveats about introducing assumptions into the model as all the ideas discussed in the previous section.

The results of evolving robots for the predictable change task are an example of some of the dangers of this approach: an assumption was made about the kind of controller that would be good at this task, and the fitness function used to encourage such controllers to develop, but in fact this requirement was not useful and produced robots less capable of performing well.

There are many possibilities for what controller attributes might be selected on, and examples include high connectivity (which was used in some of the preliminary work for the growth and pruning controllers, and found to be very useful when connectivity is low), lower connectivity, basic functionality (e.g. connecting sensors to motors in part one), sensory-motor loops, scale-free networks and symmetrical controllers or behaviour.

It is difficult to know which attributes might be useful to select for until the evolutionary process has been observed, and it becomes clear how it must be altered to avoid certain pitfalls and climb the slopes. This emphasises the need for researchers to expect to adapt continually the models and requirements, responding to the issues which only become clear when the evolutionary process is better understood.

The use of selection for connecting sensors to motors in chemical growth controllers provided clear advantages, and selection for high connectivity in growth and pruning controllers showed definite advantages in situations where the robot's lifetime began with fewer than 100% of its synapses in place, and despite the fact that the predictable change task with selection for change in the controller did not result in better controllers, it remains the view of the author that selection for properties of the internal structure of the controller along with external behaviour will be crucial in the successful evolution of two-tier structural change developmental controllers in the future.

There is a symmetry here between the desire for two-tier structure in the controller and the need for a two-tier fitness function to encourage that structure to form. It seems that there is a need to re-learn the lesson that evolution only evolves the minimal necessary solution to any problem, and if it is to be forced to produce complex multi-level structures it must be pushed and pulled in the right directions at all levels, not merely the behavioural one.

Summary

It is possible that the use of development to generate significantly better controllers than more fixed schemes is simply beyond the current level of understanding in evolutionary robotics. In general in this work it was found that developmental controllers are harder to evolve than fixed ones in situations where the development may be of no use, and do not perform significantly better until the tasks become quite complex. Perhaps when the field moves on to studying much more complex tasks the use of development will need to be commonplace, but for the moment its use needs to be justified to ensure it is the right choice.

It is notable that developmental controllers were found to be more robust in certain situations than more fixed ones when perturbed by unexpected disruptions. It appears that in some circumstances, starting from a small or random starting point and generating structural elements can

[2]The author is grateful to Ezequiel Di Paolo for suggesting this idea.

provide behavioural robustness to unexpected disruptions 'for free' even when the robots are not evolved to deal with such situations.

If in the future a truly robust developmental process can be designed that exhibits canalisation without being inflexible, brittle or unevolvable, it may be possible that this process will allow us to evolve more complex systems due to the scope for modularity, symmetry and adaptability, but given the current state of understanding, developmental systems add a new level of complexity to the process, and often the potential gains are difficult to realise. The complexity added is not simply in terms of a numerical increase in the number of variables evolved (since in principle this does not make evolution harder if good solutions are distributed in the same way) but an increase in the number of levels of abstraction between genotype and behaviour, which can easily lead into local fitness maxima from which it is difficult to escape.

In the analyses of evolved robots, regulatory and meta processes were found to be crucially important (particularly see the timing of the growth of a neuron in section 7.2.3), but they are also the biggest cause of indirection of the link between fitness and genotype, so need to be handled very carefully. It may be that it is vital to allow regulatory genes to exist since they seem to be very powerful. On the other hand, allowing them too much power may lead to many evolutionary blind alleys where the developmental or regulatory process controls the robot instead of building a lower tier of dynamics which performs the control. This result fits with the ideas of Gould (1977), who discusses the importance of regulatory genes in the developmental process, especially in their control of the timing of developmental events.

Thus, contrary to the expectations of the author, development is not the easy solution to the 'glass ceiling' of complexity which appears to have been hit in the domain of evolving CTRNNs, but it may one day become the difficult solution to a difficult problem. However, it is worth noting that the idea that this type of process will provide a solution is still based on intuition rather than solid fact. The work in this thesis provides a little evidence for this idea, since the developmental controllers perform better on more complex tasks, but it is still possible that development is a necessary extra problem that nature has managed to work around, rather than a good way to gain scalability or adaptability. Artificial evolution may need to find its own ways to gain these advantages and generate more complex behaviour.

It is hoped that the ideas presented above will provide, along with the specifics of the design of controllers and experiments, a set of tools in terms of ways of thinking, techniques and 'rules of thumb' which may be useful to researchers looking into the area of evolving developmental systems.

10.2 Exploring open questions

In this section, each open question enumerated in section 4.2 is examined in the light of the exploratory evidence collected by experimentation.

1. May developmental systems be evolved to generate simple behaviours using today's methods?
It has been shown through all of the experiments described in this thesis that it is feasible to evolve controllers containing explicit developmental mechanisms to perform simple behaviours. However, as may be seen especially in the work of part 1, it does appear to be more difficult to

evolve these controllers than those without this addition. Some possible explanations for this, and possible solutions, were discussed in the previous section.

2. Can developmental systems handle predictable changes in required behaviour better than non-developmental ones?

No evidence has been found to support the idea that developmental controllers may be evolved to perform better in situations requiring predictable changes in behaviour. The predictable change experiment (section 9.1.3) showed no statistically significant difference in performance between the three controller types growth and pruning, plastic and fixed.

The performance of both behaviours was well within the capabilities of the fixed controllers, and the addition of the ability to alter network properties and structure did not lead to improvements in final fitness achieved. Examination of the behaviour of the robots suggests that the fixed controllers did not undergo significant change in dynamics at the point in which a different behaviour was required, but simply embodied a strategy which responded correctly to the two different stimuli of being enclosed in a corridor and being in open space. It is possible that if a task were contrived which did not allow for such a reactive strategy (since the immediate sensory input was similar in both parts of the task), the potential benefits of the addition of a developmental mechanism might have been seen.

3. Will developmental systems selected for change in the controller at the time at which behavioural change is required evolve more successfully to exhibit that change?

The results of the predictable change task (section 9.1.3) provided no evidence to support the idea that a modified fitness function encouraging controller change at the time in a robot's lifetime when behavioural change was required could be beneficial to the final fitness achieved. In fact, the robots evolved under these conditions were shown to achieve a lower fitness than those evolved under normal conditions.

It appears that many good strategies for this task involved no controller change, or only a small amount of controller change, and by 'distracting' evolution, causing it to pursue strategies involving large amounts of change, the fitness achieved was reduced.

4. May developmental systems be evolved to perform tasks that are too complex for any non-developmental system?

The growth and pruning and plastic controllers did not achieve higher fitness than fixed controllers in most of the tasks. They did achieve higher fitness in two experiments (learning, section 9.1.6 and re-learning, section 9.1.7), showing that the additional mechanisms provided were beneficial, but since both controller types had more free parameters and were more easily able to preserve state over much longer timescales (an ability particularly relevant to these tasks), it is not possible to come to any conclusion as to whether the explicitly developmental character of the added dynamics was relevant to the success achieved. In fact, it is likely that a fixed controller containing a larger number of neurons (giving it a comparable number of free parameters) and with long time constants would have been able to achieve fitnesses comparable with those achieved by the growth and pruning and plastic controllers. This conjecture is supported by the high fitness achieved evolving such a controller in the T Maze task in order to test robustness (section 9.1.9).

It is a matter for speculation as to whether there are tasks which would be better addressed by

controllers containing explicit developmental systems, but in the tasks chosen here it appears that development provides little benefit.

In order to explore fully this question, one would have to examine the space of all possible non-developmental controllers and show that there was a developmental controller which outperformed all of them in a particular task. A first step in this process is offered here through the examination of a single example of each type of controller.

5. Are developmental controllers more capable of performing learning tasks than non-developmental ones?

As outlined in the above section, the two learning experiments (sections 9.1.6 and 9.1.7) showed higher fitness for the developmental controllers, but this result may be explained by the greater complexity and long temporal scales of these controllers, rather than their developmental abilities. Thus little or no evidence was found to support the idea that developmental controllers are more capable of performing learning tasks than non-developmental ones.

6. Are developmental controllers more able to produce different behaviours depending on the type of environment with which they are faced?

The flexibility experiment (section 9.1.8) showed that in the task used controllers with additional developmental capacities performed no better in terms of final fitness than fixed controllers. No evidence was found to support the idea that developmental controllers are more able to produce different behaviours in different environments.

Since the task chosen was found to be well within the capabilities of the fixed controllers, despite the fact that two different behaviours were required depending on the circumstances, there was no real opportunity for the developmental controllers to show the benefits (if any) of the ability to generate entirely different behaving controllers under different circumstances. It is possible that if each of the two tasks were complex enough to use the full capacity of the fixed controllers, the developmental controllers might have shown themselves capable of performing better, but in this case, as in that of the question involving learning, there is no guarantee even if that were so that larger fixed controllers with longer time constants would not perform equally as well.

7. Are developmental controllers more robust to previously unencountered change than non-developmental ones?

The robustness experiments (section 9.1.9) provide evidence to suggest that developmental controllers may be more robust than their more static counterparts in some circumstances. The idea behind this question is that since the robot must construct its controller at the beginning of its lifetime, and since that construction process may be affected by the robot's experience, a coupling might be formed between the successful performance of behaviour and the successful formation of the controller. This coupling could lead to the modification of the controller structure generated in cases of previously unencountered change, producing stable behaviour despite that change.

Such a coupling appears to have been generated in the experiments performed, causing the developmental controllers to exhibit significantly greater robustness than the small and medium fixed controllers, plastic controllers, and large fixed controllers with comparable complexity and temporal range. An interesting prediction from this result is that, since this coupling already occurs in some cases, it must be relatively easy to achieve, and therefore it seems likely that if robustness

were selected for in developmental controllers, encouraging the coupling to occur in more cases, they might respond better to such selection than fixed controllers, widening further the fitness gap between such controllers and fixed ones, even in the case where the fixed controllers, too, were selected for robustness. This is an interesting potential avenue for future research.

8. Are developmental controllers evolved to be robust to noise in the developmental process more robust to previously unencountered change than those evolved in more predictable conditions?

It was expected that reducing the reliability of individual elements (by allowing spontaneous growth and death of synapses with the growth and pruning 95* controllers, in the robustness experiment of section 9.1.9) might have encouraged more flexible, holistic strategies where development produced redundant mechanisms that continue to perform the correct behaviour even in disrupted conditions, but although moderately successful controllers for simple tasks were evolved, the expected robustness was not observed - these controllers were little more robust than fixed or plastic networks, if at all, and their degraded performance even in the unaltered task prevented them from entering consideration as a viable alternative.

No evidence was found to support the idea that robustness to unreliability or noise in a controller encourages robustness in behaviour. It is possible that the noise introduced was simply too extreme to be overcome by the evolutionary process in this case. It would be interesting to examine the effects on robustness of introducing a range of different noise levels into the developmental process, and this might be an interesting route for further work.

9. Will controllers whose development involves mainly guided pruning of structure be more successful than those whose development involves mainly guided growth?

By comparing the final fitness of growth and pruning controllers evolved with differing levels of synapse growth and death (section 9.1.10), it was found, in one of the two experiments, that in controllers some of whose synapses are dead at the beginning of the lifetime, synaptic growth is important, whereas death is relatively unimportant. Thus there is some reason to suppose that the idea that controllers involving mainly guided pruning of structure perform more successfully than those involving mainly guided growth may not be correct. This provides very weak support for the constructivist view of brain development which emphasises the role of guided growth of brain structure.

The constructivist viewpoint suggests that brain structures in real animals are generated through guided growth of brain material. The first experiment in the synapse growth verses death comparison, using growth and pruning 90 controllers and a simple phototaxis task, offers some encouragement for the holders of this view since it shows that in this situation guided, experience-dependent growth of artificial synapses can result in useful controllers being generated. Furthermore, it shows that variation in the levels of synapse death do not appear to affect the outcome of evolution in terms of the mean fitness of the best generated individuals.

It appears that in both the phototaxis and T maze experiments synapse death was not very useful for generating controllers for robots. This could be for a number of reasons: first, it is possible to recreate the effect of synapse death by reducing connection strength to zero, so any robot requiring the removal of a particular synapse may simply use a plasticity rule to reduce its strength instead. Second, it was observed that many ‘useless’ synapses were not removed during development: synapses that appeared to play no part in the behaviour, and whose strengths

varied widely over different lifetimes, even in identical environments, often did not die. This may be because if such synapses are allowed to have uncontrolled, 'random' values, their effects are generally balanced by those of other random synapses, and the overall effect on behaviour is minimal. Robots certainly seem able to behave effectively in the presence of such synapses.

This presents an interesting question for the proponents of selectionism: is the removal of brain material useful for generating behaviour in animals? There are several possible ways in which it might be useful. In situations involving competition between synapses for limited resources, useful structures may emerge more frequently than in the situation simulated in these tests, where synapse death did not imply an extra capacity for the growth of another synapse. Also, removal of brain elements in real animals may simply arise from a need to save energy.

This work suggests some interesting further lines of enquiry. It would be useful to investigate the question discussed above concerning where death of brain elements could be useful. One idea, that competition between synapses produces useful structures, could be explored by comparing robots whose synapses compete for finite resources (perhaps limiting the total strength of all the synapses feeding into a neuron) with those whose synapses may take any strength. If the competitive system results in higher mean fitnesses, there may yet be reason to explore further the ideas behind the question which heads this section.

Another line for future work lies in modelling more explicitly the systems proposed by the two viewpoints. This could mean a system in which all synaptic growth is spontaneous and random, and all death is guided by the genotype and neuron activity (approximating the situation suggested by selectionists) being compared with a system in which all growth is guided and death is either random, or also guided (approximating the situation suggested by constructivists). The case of both being guided is the one which has been treated in this thesis.

Of course, all of these tests, and those proposed, at best may only show in principle which types of system can produce useful behaviours, and offer no guidance as to what actually happens in real animals. If one way of generating controllers is shown to be vastly superior to another, this may provide some guidance as to what might be expected to have evolved in nature, but as with all this work, the comparisons being drawn between natural and artificial systems are extremely abstract and general, and many assumptions must be made in order to apply the findings in one situation to the realm of the other.

10.3 Further discussion

The issue of where development may be useful in evolutionary robotics is discussed in section 10.3.1, and the specific conclusions that may be drawn from the failure of the growth and pruning 90 controllers to evolve good solutions are discussed in section 10.4.

10.3.1 Where is development useful?

In practical terms, for the artificial evolution of robots capable of performing simple behaviours, where might it be useful to incorporate a developmental control mechanism? Certainly there are some situations in which such mechanisms may be beneficial.

The developmental controllers studied in parts one and two were successfully evolved to perform relatively complex tasks, sometimes evolving to a greater competence than fixed controllers.

The first observation is that although in some cases these developmental systems appeared difficult to evolve, the barriers were not insurmountable: despite the reservations described in section 10.1.2, it is practical to study simple developmental systems using contemporary genetic algorithm techniques and technology.

The chemical-guided growth networks performed admirably on the orientation and discrimination tasks, but in the most difficult task they showed a tendency to over-specialise. However, given strong constraints on the controller and fitness function, it was possible to evolve robust and elegant solutions for the multiple discrimination task. This task requires a complex behaviour, but one that involves responding to immediate sensory input rather than using internal state, not a situation that might be expected to show the benefits of a controller capable of long term structural change. Indeed, this task evolves more easily with a simpler, more reactive controller, but the developmental system is capable of generating an appropriate controller under the right circumstances.

The problems of complexity, indirection, over-specialisation and building structure from a blank slate experienced in part one were addressed in the design of the growth and pruning controllers used in part two. These controllers (in their 'Growth and Pruning 100' form) were successfully evolved to perform a variety of tasks, starting with a simple phototaxis task, moving through corridor following to tasks involving maintenance of internal state.

In the simpler tasks there was no clear advantage for developmental controllers, and the disadvantages of needing to grow synapses before being able to guarantee their presence meant that the growth and pruning 90 controllers performed poorly on most of the tasks. Also, in some of the more complex environments, notably the flexibility and predictable change tasks, the fixed controllers performed very well, and no significant difference in fitness was found between them and the developmental controllers.

In the re-learning task, the plastic and developmental controllers significantly outperformed the fixed ones, and the advantages of the potential for structural change of the developmental controllers, which had slowed their progress in simpler tasks, enabled them to perform as well as the plastic controllers in this more complex task. However, since the fixed controllers were limited to a smaller number of free parameter and shorter timescales, this conclusion does not suggest that fixed networks, given longer timescales and more free parameter, would not perform as well as the growth and pruning and plastic controllers. In this case, the fixed controllers may well evolve dynamics which also might be described as 'developmental.'

If the trend of growth and pruning controllers' performance improving with task complexity were found to continue, it might be possible that even more complex tasks, involving long-term storage of state which is subject to change under certain circumstances, would evolve to a higher fitness with the growth and pruning controllers than with the plastic. However, it is worth noting that the growth and pruning controllers that were most successful were the growth and pruning 100 variety, which are quite similar to plastic networks - if no growth or death occurs they are identical - and it may be that the ability to remove and add synapses is not useful for performing this kind of task. The analysis made of a robot performing the re-learning task shows that, at least in this case, the capacities to remove and add synapses were not used a great deal, and where they were used they did not serve a particularly complex purpose. Circumstances under which growth

and death may be more useful, for example in competitive situations, are discussed in section 10.2.

The developmental controllers performed better than fixed ones in some of the tests of robustness to disruptions. In this case it was shown that even fixed controllers with an equivalent number of free parameters and the capacity to have long term dynamics did not behave as robustly as growth and pruning controllers. If this feature can be exploited it will be extremely useful for designing robots which need to preserve stable behaviours in the face of unexpected changes in their bodies or environments.

It was surprising how well the fixed controllers evolved to perform the tasks, even those involving high levels of adaptivity such as the predictable change and flexibility tasks. The plastic controllers also performed well above expected levels. It seems that structural change (in terms of growth and death of synapses) is not crucial to perform tasks at this complexity level.

A further potential advantage of the developmental and plastic controllers was that they evolved much more consistently than the fixed ones in several of the tasks. The histograms of final fitnesses found for the fixed controllers showed that they were often distributed widely, with some runs reaching very high fitness and others failing to evolve adequate solutions. In contrast, the fitness scores of the plastic and developmental controllers were more consistent, clustered near to the higher fitness end of the distribution. In a situation where it was important to evolve an adequate solution using a single evolutionary run, a developmental or plastic controller might well be a good choice even for a simple reactive task. However, it is likely that part of the explanation for this phenomenon is the larger number of free parameters under evolutionary control in the two more developmental controller types. It is often observed that increasing the number of free parameters can tend to smooth the path through evolutionary space. Thus, using larger fixed controllers might have the same effect in this case.

10.4 Growth and Pruning 90

The 'Growth and Pruning 90' controllers failed to reach the same fitness as the other controller types in several of the tasks. These controllers involved a greater developmental burden since each synapse had a good chance of being dead when the robot's lifetime began. Thus in order to behave in a consistent way the robot needed to grow any missing synapses before modifying their weights as required. This process introduces a great deal of noise into the fitness evaluation, since, for example, if the robot begins its life with a crucial synapse connecting a sensor to a neuron missing, and there is no current sensory input, it may be difficult to guarantee that the appropriate sensory input will be experienced, which is needed to trigger growth from a growth rule that is not 100% optimised to grow under all sensory circumstances. In this case the robot is unlikely to respond correctly to the sensory input for some time, until it experiences the right situation for that synapse to grow.

It is possible that if these controllers had been evaluated over more lifetimes this noise factor might have been reduced sufficiently to allow more effective evolution. If this were the case some of the advantages of re-constructing the controller in every lifetime might have been realised, including a possible greater flexibility to variations in the environment.

The failure in many tasks of the growth and pruning 90 controllers to perform as well as the plastic and fixed controllers is a further demonstration of the difficulty involved in building up

structure from fewer to more elements. It seems likely that the problem of bootstrapping in this situation, outlined in detail in section 8.1 will continue to be a major issue for those wishing to evolve developmental systems.

10.5 Conclusions and future directions

This work was undertaken in order to develop experimental tools to be used in the study of development, and to answer some specific questions about developmental phenomena.

The unexpected difficulty encountered in evolving controllers capable of simple behaviours means that the tools developed cover not only specific controllers types and evolutionary techniques, but also explanations and ideas which may help future researchers to circumvent some of the problems and challenges encountered.

Most of the questions addressed warrant a great deal of further investigation, but it is hoped that this work demonstrates that these questions are open to investigation using current techniques and methodologies. No doubt the tentative conclusions reached here will quickly be revealed as naive as best, but they may provide stepping-off points for others to continue learning.

Several important avenues for the continuation of this work have been discussed in the preceding sections, including some concrete proposals for further experiments. It is hoped that if these avenues are followed, the goals of this work may be further advanced.

It is possible that in order to reap all of the benefits seen in biological developmental systems research must move beyond the information-transfer paradigm of neo-Darwinian genetic algorithm research altogether. The ideas discussed in section 3.2 about the need to remove the conceptual separation between genotype and phenotype may need to be taken on board into the models that are used, and the problem of evolving from a random starting point rather than from a point where the accumulated benefits of a canalised developmental system are already in place need to be addressed.

How to move beyond these methods and paradigms is an extremely difficult question, and not one that is addressed in this thesis, but it is a question that must be addressed if the goal of understanding development to any extent is ever to be reached. Possible directions which may be useful are contained in the work of Macinnes (2005), who is looking at evolving 'functional circles' (self-extinguishing sensory-motor-behaviour loops) rather than more abstract fitness measures, and has also developed successful theory and methods for incremental evolution which may eventually provide the field with the capacity to deal with evolutionary histories rather than isolated genetic algorithm runs.

The most important principle in the design of developmental systems discovered through this work is that evolution and development appear to work better by optimising pre-existing complex structures, even if they are highly disordered, than working to construct new structures needed for a particular purpose. This principle has been put into practice and shown to have a strong positive effect on the evolution of developmental systems. It is hoped that this idea will be developed in the future into a well-defined theory which may be rigorously tested, and, if found to be correct, applied to improve the design of artificial developmental systems in the future.

Bibliography

Ackley, D. and Littman, M. (1991). Interaction between learning and evolution. In Langton, C., Taylor, C., Farmer, J., and Rasmussen, S., editors, *Artificial Life II*, pages 487–509.

Astor, J. and Adami, C. (1998). Development and evolution of neural networks in an artificial chemistry. In Wilke, C., Altmeyer, S., and Martinetz, T., editors, *Proc. of 3rd German Workshop on Artificial Life*, pages 15–30. Verlag Harri Deutsch.

Beer, R. (1996). Toward the evolution of dynamical neural networks for minimally cognitive behavior. In Maes, P., Mataric, M., Meyer, J., Pollack, J., and Wilson, S., editors, *From Animals to Animats 4*, pages 421–429. MIT press.

Beer, R. (2000). Dynamical approaches to cognitive science. *Trends in Cognitive Sciences*, 4(3):91–99.

Beer, R. (2004). The dynamics of active categorical perception in an evolved model agent. *Adaptive Behaviour*, 11(4):209–243.

Belew, R. (1992). Interposing an ontogenetic model between genetic algorithms and neural networks. *NIPS*, pages 99–106.

Brooks, R. (1991). Intelligence without reason. In Myopoulos, J. and Reiter, R., editors, *Proceedings of the 12th International Joint Conference on Artificial Intelligence (IJCAI-91)*, pages 569–595, Sydney, Australia. Morgan Kaufmann publishers Inc.: San Mateo, CA.

Cangelosi, A. (1999). Heterochrony and adaptation in developing neural networks. In Banzhaf, W. and et al., editors, *Proceedings GECCO99: Genetic and Evolutionary Computation Conference*, pages 1241–1248, Orlando (USA).

Cangelosi, A., Nolfi, S., and D., P. (2003). *On Growth, Form, and Computers*, chapter Artificial life models of neural development, pages 339–354. Academic Press, London UK.

Cangelosi, A., Parisi, D., and Nolfi, S. (1994). Cell division and migration in a 'genotype' for neural networks. *Network*, 5:497–515.

Changeux, J.-P. (1997). Variation and selection in neural function. *Trends in Neurosciences*, 20(7):291–293.

Chien, S., Yang, Z., and Hou, E. (2001). Genetic algorithm approach for transit route planning and design. *Journal of Transportation Engineering, ASCE*, 127(3):200–207.

Cliff, D., Harvey, I., and Husbands, P. (1996). Artificial evolution of visual control systems for robots. In Srinivisan, M. and Venkatesh, S., editors, *From Living Eyes to Seeing Machines*. Oxford University Press.

Dale, K. (2002). How bees might do it: simple evolved models of navigation in an uncertain world. In *EPSRC/BBSRC International Workshop on Biologically-Inspired Robotics: The Legacy of W. Grey Walter*, pages 118–125.

Dellaert, F. (1995). Toward a biologically defensible model of development. Master's thesis, Case Western Reserve University, Dept. of Computer Engineering and Science.

Dellaert, F. and Beer, R. (1994a). Co-evolving body and brain in autonomous agents using a developmental model. Technical Report CES-94-16, Department of Computer Engineering and Science, Case Western Reserve University, Cleveland, OH 44106.

Dellaert, F. and Beer, R. (1994b). Toward an evolvable model of development for autonomous agent synthesis. In *Artificial Life IV*. MIT Press Cambridge.

Dellaert, F. and Beer, R. (1996). A developmental model for the evolution of complete autonomous agents. In Maes, P., Mataric, M., Meyer, J., Pollack, J., and Wilson, S., editors, *SAB '96 Conference Proceedings: From Animals to Animats 4: Proceedings of the Fourth International Conference on Simulation of Adaptive Behavior*, Cambridge, MA. The MIT Press/Bradford Books.

Di Paolo, E. (2003). Evolving spike-timing-dependent plasticity for single-trial learning in robots. *Philosophical Transactions: Mathematical, Physical & Engineering Sciences*, 361(1811):2299–2319(21).

Di Paolo, E. A. (2000). Homeostatic adaptation to inversion of the visual field and other sensorimotor disruptions. In *Proceedings of SAB 2000*, pages 440–449.

Di Paolo, E. A., Noble, J., and Bullock, S. (2000). Simulation models as opaque thought experiments. In *Artificial Life VII: The Seventh International Conference on the Simulation and Synthesis of Living Systems*, pages 497–506, Reed College, Portland, Oregon, USA.

Eggenberger, P. (1997). Creation of neural networks based on developmental and evolutionary principles. In *International Conference on Artificial Neural Networks ICANN'97*, pages 337 – 342, Lausanne, Switzerland.

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*, 36:149–169.

Elman, J. (1993). Learning and development in neural networks: The importance of starting small. *Cognition*, 48:71–99.

Elman, J., Bates, E., Johnson, M., Karmiloff-Smith, A., Parisi, D., and Plunkett, K. (1996). *Rethinking Innateness: a Connectionist Perspective on Development*. Cambridge Ma.: MIT Press.

Ewert, P. (1930). A study on the effect of inverted retinal stimulation upon spatially coordinated behavior. *Genetic Psychological Monographs*, 7:3–4.

Fernando, C. (2002). An evolutionary robotic model of instrumental and classical learning. Master's thesis, School of Cognitive and Computing Sciences, University of Sussex, Brighton, UK.

Ficici, S. G., Watson, R. A., and Pollack, J. B. (1999). Embodied evolution: A response to challenges in evolutionary robotics. In Wyatt, J. and Demiris, J., editors, *Eighth European Workshop on Learning Robots*, pages 14–22.

Fleischer, K. and Barr, A. H. (1993). A simulation testbed for the study of multicellular development: The multiple mechanisms of morphogenesis. *Artificial Life*, 3:389–416.

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, *Proceedings of SAB 4*, pages 402–410. MIT Press-Bradford Books.

Floreano, D. and Mondada, F. (1998). Evolutionary neurocontrollers for autonomous mobile robots. *Neural Networks*, 11:1461–1478.

Funahashi, K. and Nakamura, Y. (1993). Approximation of dynamical systems by continuous time recurrent neural networks. *Neural Networks*, 6(6):801–806.

Goldberg, D. (1986). The genetic algorithm: Who, how, and what next? *Adaptive and Learning Systems*.

Goodwin, B. (1994). *How the Leopard Changed Its Spots : The Evolution of Complexity*. Weidenfield and Nicholson London.

Gould, S. J. (1977). *Ontogeny and Phylogeny*. Harvard University Press.

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.

Gruau, F. (1995). Automatic definition of modular neural networks. *Adaptive Behaviour*, 3(2):151–183.

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

Harp, S. A., Samad, T., and Guha, A. (1989). Towards the genetic synthesis of neural networks. In *Proceedings of the third international conference on genetic algorithms*, pages 360–369. Morgan Kaufmann.

Harvey, I. (2005). Continuous-time recurrent neural networks do not 'lack plasticity'. Personal Communication.

Hinton, G. and Nowlan, S. (1987). How learning guides evolution. *Complex Systems*, 1:495–502.

Holland, J. (1975). *Adaptation in Natural and Artificial Systems*. MIT Press.

Husbands, P., Harvey, I., Cliff, D., and Miller, G. (1994). The use of genetic algorithms for the development of sensorimotor control systems. In Gaussier, P. and Nicoud, J.-D., editors, *From Perception to Action*, pages 110–121. IEEE Computer Society Press, Los Alamitos CA.

Husbands, P., Smith, T., Jakobi, N., and O'Shea, M. (1998). Better living through chemistry: Evolving gasnets for robot control. *Connection Science Special Issue: BioRobotics*, 10(3-4):185–210.

Ivanco, T. and Greenough, W. (2000). Physiological consequences of morphologically detectable synaptic plasticity: potential uses for examining recovery following damage. *Neuropharmacology*, 39:765–776.

Jakobi, N. (1995). Harnessing morphogenesis. Technical Report 423, University of Sussex, School of Cognitive and Computing Sciences, Brighton BN1 9QH, UK.

Jakobi, N. (1997). Evolutionary robotics and the radical envelope-of-noise hypothesis. *Adaptive Behavior*, 6(2):325–368.

Jakobi, N. (1998). *Minimal Simulations for Evolutionary Robotics*. PhD thesis, University of Sussex.

Karmiloff-Smith (1992). *Beyond Modularity: A Developmental Perspective on Cognitive Science*. MIT Press.

Kelso, J. A. S. (1995). *Dynamic Patterns*. MIT Press.

Kempter, R., Gerstner, W., and van Hemmen, J. L. (1999). Hebbian learning and spiking neurons. *Physical Review E*, 59:4498–4514.

Kitano, H. (1990). Designing neural networks using genetic algorithm with graph generation system. *Complex Systems*, 4:461–476.

Kitano, H. (1994). A simple model of neurogenesis and cell differentiation based on evolutionary large-scale chaos. *Artificial Life*, 2(1):79–99.

Kodjabachian, J. and Meyer, J. (1998). Evolution and development of modular control architectures for 1-d locomotion in six-legged animats. *Connection Science*, 10:211–237.

Kolb, B., Forgie, M., Gibb, R., Gorny, G., and Rowntree, S. (1998). Age, experience and the changing brain. *Neuroscience and Biobehavioural Reviews*, 22(2):143–159.

Koza, J. R. (1991). Genetic evolution and co-evolution of computer programs. *Artificial Life II*, pages 603–629.

Lindenmayer, A. and Rozenberg, G. (1976). *Automata, languages, development*. North-Holland, Amsterdam.

Macinnes, I. A. (2005). *DPhil Thesis (in preparation)*. PhD thesis, CCNR, University of Sussex.

McIlhagga, M., Husbands, P., and Ives, R. (1996). A comparison of search techniques on a wing-box optimisation problem. In Voigt, H.-M., Ebeling, W., Rechenberg, I., and Schwefel, H.-P., editors, *Proceedings of the Fourth Conference on Parallel Problem Solving from Nature: PPSN IV*, pages 614–623. Springer-Verlag.

Mitchell, M. (1996). *An Introduction to Genetic Algorithms*. MIT Press.

Nolfi, S. (2002). *Handbook of brain theory and neural networks, Second Edition*, chapter Evolution and Learning in neural networks, pages 415–418. MIT Press.

Nolfi, S. and Floreano, D. (2000). *Evolutionary Robotics: The Biology, Intelligence, and Technology of Self-Organizing Machines.* Cambridge, MA: MIT Press/Bradford Books.

Nolfi, S., Miglino, O., and Parisi, D. (1994). Phenotypic plasticity in evolving neural networks. In *Proceedings of PerAc 1994*, pages 146–157. IEEE Computer Society Press.

Nolfi, S. and Parisi, D. (1992). Growing neural networks. Technical report, Institute of Psychology, Rome.

Nolfi, S. and Parisi, D. (1995). Evolving artificial neural networks that develop in time. In Morn, F., Moreno, A., Merelo, J., and Chacn, P., editors, *Advances in Artificial Life. Proceedings of the Third European Conference on Artificial Life*, pages 353–367. Berlin: Springer-Verlag.

Nolfi, S. and Parisi, D. (1996). Learning to adapt to changing environments in evolving neural networks. *Adaptive Behavior*, 5(1):75–98.

Oyama, S. (2000). *The Ontogeny of Information: Developmental Systems and Evolution*. Duke University Press.

Purves, D. (1994). *Neural Activity and the Growth of the Brain*. Cambridge, England: Cambridge University Press.

Purves, D., White, L., and Riddle, D. (1996). Is neural development darwinian? *Trends in Neurosciences*, 19:460–464.

Quartz, S. (1999). The constructivist brain. *Trends in Cognitive Sciences*, 3(2):48–57.

Quartz, S. and Sejnowski, T. (1997). The neural basis of cognitive development: a constructivist manifesto. *Behavioral and Brain Sciences*, 4(20):537–596.

Rao, R. P. and Sejnowski, T. J. (2001). Spike-timing-dependent hebbian plasticity as temporal difference learning. *Neural Comput.*, 13(10):2221–2237.

Sizonenko, P. (2003). *Reproductive Health: Second Postgraduate Course for Training in Reproductive Medicine and Reproductive Biology - Human Sexual Differentiation*. Faculty of Medicine, University of Geneva and Special Programme of Research, Development and Research Training in Human Reproduction, World Health Organization.

Slocum, A., Downey, D., and Beer, R. (2000). Further experiments in the evolution of minimally cognitive behavior: From perceiving affordances to selective attention. In Meyer, J., Berthoz, A., Floreano, D., Roitblat, H., and Wilson, S., editors, *From Animals to Animats 6: Proceedings of the Sixth International Conference on Simulation of Adaptive Behavior*, pages 430–439. MIT Press.

Sporns, O. (1997). Variation and selection in neural function. *Trends in Neurosciences*, 20(7):291.

Stratton, G. (1896). Some preliminary experiments on vision without inversion of the retinal image. *Psychological Review*, 3:611–617.

Stratton, G. (1897). Vision without inversion of the retinal image. *Psychological Review*, 4:361–360.

Tuci, E., Harvey, I., and Quinn, M. (2002a). Evolving integrated controllers for autonomous learning robots using dynamic neural networks. In *Proceedings of The Seventh International Conference on the Simulation of Adaptive Behavior (SAB'02)*, pages 282–291, Edinburgh, UK. MIT Press.

Tuci, E., Quinn, M., and Harvey, I. (2002b). Evolving fixed-weight networks for learning robots. In Fogel, D., El-Sharkawi, M., Yao, X., Greenwood, G., Iba, H., Marrow, P., and Shackleton, M., editors, *Proceedings of the 2002 Congress on Evolutionary Computation*, pages 1970–1975, Honolulu, Hawaii. IEEE Press.

Urzelai, J. and Floreano, D. (2000). Evolutionary robotics: Coping with environmental change. In Whitley, D., Goldberg, D., Cantu-Paz, E., Spector, L., Parmee, I., and Beyer, H.-G., editors, *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2000)*, pages 941–948, Las Vegas, Nevada, USA. Morgan Kaufmann.

Vaario, J. (1991). Developing neural networks through growth. In *Proceedings of the 5th Annual Conference of JSAI (Japanese Society for Artificial Intelligence)*, Tokyo.

van Praag, H., Schinder, A. F., Christie, B. R., Toni, N., Palmer, T. D., and Gage, F. H. (2002). Functional neurogenesis in the adult hippocampus. *Nature*, 415(6875):1030–1034.

Vickerstaff, R. and Di Paolo, E. (2005). Building neural models of path integration using artificial evolution, in preparation. *Journal of Experimental Biology*.

Waddington, C. H. (1975). *The Evolution of an Evolutionist*. Cornell University Press.

Wagner, G. and Altenberg, L. (1996). Complex adaptations and the evolution of evolvability. *Evolution*, 50(3):967–976.

Wilson, S. (1987). The genetic algorithm and biological development. In Grefenstette, J., editor, *Genetic Algorithms and Their Applications: Proceedings of the Second International Conference on Genetic Algorithms*, pages 247–251. Lawrence Erlbaum Associates.