Pareto Multi-Objective Evolution of Legged Embodied Organisms

Jason T.W. Teo
B.Comp.Math. (University of Western Australia, WA, Australia)
M.Info.Tech. (Charles Sturt University, NSW, Australia)

submitted in partial fulfillment of the requirements
for the D.IT degree at the
School of Computer Science
University of New South Wales
Australian Defence Force Academy Campus

A thesis submitted for the degree of Doctor of Information Technology.

2003

Abstract

The automatic synthesis of embodied creatures through artificial evolution has become a key area of research in robotics, artificial life and the cognitive sciences. However, the research has mainly focused on genetic encodings and fitness functions. Considerably less has been said about the role of controllers and how they affect the evolution of morphologies and behaviors in artificial creatures. Furthermore, the evolutionary algorithms used to evolve the controllers and morphologies are pre-dominantly based on a single objective or a weighted combination of multiple objectives, and a large majority of the behaviors evolved are for wheeled or abstract artifacts.

In this thesis, we present a systematic study of evolving artificial neural network (ANN) controllers for the legged locomotion of embodied organisms. A virtual but physically accurate world is used to simulate the evolution of locomotion behavior in a quadruped creature. An algorithm using a self-adaptive Pareto multi-objective evolutionary optimization approach is developed.

The experiments are designed to address five research aims investigating: (1) the search space characteristics associated with four classes of ANNs with different connectivity types, (2) the effect of selection pressure from a self-adaptive Pareto approach on the nature of the locomotion behavior and capacity (VC-dimension) of the ANN controller generated, (3) the efficiency of the proposed approach against more conventional methods of evolutionary optimization in terms of computational cost and quality of solutions, (4) a multi-objective approach towards the comparison of evolved creature complexities, (5) the impact of relaxing certain morphological constraints on evolving locomotion controllers.

The results showed that: (1) the search space is highly heterogeneous with both rugged and smooth landscape regions, (2) pure reactive controllers not requiring any hidden layer transformations were able to produce sufficiently good legged locomotion, (3) the proposed approach yielded competitive locomotion controllers while requiring significantly less computational cost, (4) multi-objectivity provided a practical and mathematically-founded methodology for comparing the complexities of evolved creatures, (5) co-evolution of morphology and mind produced significantly different creature designs that were able to generate similarly good locomotion behaviors. These findings attest that a Pareto multi-objective paradigm can spawn highly beneficial robotics and virtual reality applications.

Keywords

Artificial intelligence, artificial life, artificial neural networks, body-brain co-evolution, complexity measures, differential evolution, evolutionary algorithms, evolutionary multi-objective optimization, evolutionary artificial neural networks, evolutionary robotics, morpho-functional machines, multi-objective optimization.

Acknowledgements

First and foremost, I am truly grateful to my supervisor Dr. Hussein Abbass who has gone far beyond the call of duty in accompanying me on this amazing journey. My utmost thanks.

I am also grateful to the academics and staff of the School of Computer Science, UNSW@ADFA for their help along the way. In particular, my thanks goes to Prof. Charles Newton, Dr. Chris Lokan, Dr. Bob McKay, Dr. Frantz Clermont, Dr. Ed Lewis and Dr. Spike Barlow.

My acknowledgement also goes to the numerous anonymous referees for their constructive comments on various parts of this thesis that have been published.

To Mum and Dad for their encouragement and advice. Mum, although you had to leave so suddenly last year, I know you will cherish this accomplishment with the rest of us from above.

I wish to also thank my parents-in-law, Susan and Steven. You guys are the best!

The most thanks definitely goes to my wife Annie for her unending love, support and understanding. You are my one true inspiration.

Finally, to God for this wonderful life.

Jason Teo
Canberra, 2003.

Declaration

I hereby declare that this submission is my own work and to the best of my knowledge it contains no material previously published or written by another person, nor material which to a substantial extent has been accepted for the award of any other degree or diploma at UNSW or any other educational institution, except where due acknowledgement is made in the thesis. Any contribution made to the research by others, with whom I have worked at UNSW or elsewhere, is explicitly acknowledged in the thesis.

I also declare that the intellectual content of this thesis is the product of my own work, except to the extent that assistance from others in the project's design and conception or in style, presentation and linguistic expression is acknowledged.

Signed: _____

Date: _____

Publications

Peer-reviewed publications arising from research work conducted in this thesis are listed chronologically below (latest to earliest):

- Jason Teo and Hussein A. Abbass. "Search Space Difficulty of Evolutionary Neuro-Controlled Legged Robots", Accepted to appear in the International Journal of Knowledge-Based Intelligent Engineering Systems, Special Issue on Evolutionary Computation, July 2003.

- Jason Teo, Minh Ha Nguyen and Hussein A. Abbass. "Multi-Objectivity as a Tool for Constructing Hierarchical Complexity", Accepted to appear as a full orally-presented paper in the Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2003), Evolutionary Robotics Track, LNCS, Chicago, July 2003.

- Jason Teo and Hussein A. Abbass. "Multi-Objectivity for Brain-Behavior Evolution of a Physically-Embodied Organism", Artificial Life VIII: The 8th International Conference on Artificial Life, pp.312–318, R. Standish, M.A. Bedau, and H.A. Abbass (Eds.), ISBN 0-262-69281-3, Sydney, December 2002.

- Jason Teo and Hussein A. Abbass. "Trading-off Mind Complexity and Locomotion in a Physically Simulated Quadruped", Proceedings of the 4th Asia-Pacific Conference on Simulated Evolution And Learning (SEAL'02), Volume 2, pp. 776–780, L. Wang, K. Tan, T. Furuhashi, J. Kim, and X. Yao (Eds.), ISBN 981-04-7522-5, Singapore, November 2002.

- Jason Teo and Hussein A. Abbass. "Coordination and Synchronization of Locomotion in a Virtual Robot", Proceedings of the 9th International Conference on Neural Information Processing (ICONIP'02), Volume 4, pp. 1931–1935, L. Wang, J.C. Rajapakse, K. Fukushima, S.Y. Lee, and X. Yao (Eds.), ISBN 981-04-7524-1, Singapore, November 2002.

Contents

List of Figures

List of Tables

List of Acronyms

1D	1-Dimensional
2D	2-Dimensional
3D	3-Dimensional
ANN	Artificial Neural Network
CMM	Co-evolution of Morphology and Mind
CPG	Central Pattern Generator
EA	Evolutionary Algorithm
EANN	Evolutionary Artificial Neural Network
EMO	Evolutionary Multi-objective Optimization
GA	Genetic Algorithm
GP	Genetic Programming
HT	Hand-Tuned *
PDE	Pareto-frontier Differential Evolution
SD	Standard Deviation *
SO	Single-Objective *
SPANN	Self-adaptive Pareto Artificial Neural Network
SPDE	Self-adaptive Pareto Differential Evolution
WS	Weighted Sum *

Table I: List of acronyms used in the thesis.

* *denotes acronyms used only in tables, List of Figures and List of Tables (due to page width limitation).*

List of Mathematical Abbreviations

$\sum$	Sum of
$\lvert x \rvert$	Absolute value of x
$\neq$	Not equal to
$\approx$	Approximately equal to
$\sim$	Approximate value
$\ncong$	Not equal to, for vectors
$\prec$	Less than, for vectors
$\log_n$	Logarithm to base n
log	Logarithm to base 10
ln	Natural logarithm
∞	Infinity
$[x, y]$	Closed interval: include all points between x and y including x and y
$]x, y[$	Open interval: include all points between x and y excluding x and y
$\{\ \}$	Set notation
$\in$	Belong to
$\cup$	Union
$\subseteq$	Proper subset
ϕ	Empty set
$\lfloor x \rfloor$	Floor of x (largest integer less than or equal to x)
$\leftarrow$	Maps to
$\rightarrow$	Implies
$\Uparrow$	Maximize
$\Downarrow$	Minimize
$\lvert$	Given
iff	If and only if
$\exists$	There exists
$\nexists$	There does not exist
$\forall$	For all

Table II: List of mathematical abbreviations used in the thesis.

Chapter 1

Introduction

"The controlling Intelligence understands its own nature, and what it does, and whereon it works."

(**Marcus Aurelius, 167A.D.**)

1.1 Overview

The automatic synthesis of embodied and situated creatures through artificial evolution has become a key area of research in artificial life (Sims 1994a; Sims 1994b; Komosinski and Rotaru-Varga 2000; Bongard and Paul 2001; Komosinski and Rotaru-Varga 2001; Taylor and Massey 2001; Hornby and Pollack 2002), robotics (Mataric and Cliff 1996; Harvey, Husbands, Cliff, Thompson, and Jakobi 1997; Husbands, Harvey, Jakobi, Thompson, and Cliff 1997; Floreano 1998; Nolfi and Floreano 2000; Pollack, Lipson, Hornby, and Funes 2001; Pollack, Lipson, Ficici, Funes, and Hornby 2002), and the cognitive sciences (Dautenhahn 1996; Mataric 1997; Pfeifer and Scheier 1999; Dautenhahn 1999; Nolfi and Floreano 2002). This concept stresses the importance of studying systems that have a body and are situated in a physical environment. It also emphasizes the utilization of artificial evolution as the primary mechanism for driving the self-organization process. This approach enables artificial creatures to autonomously develop intelligent behavior through the dynamic

interactions between its body, nervous system and environment.

Research into evolving artificial creatures typically involves real-life robots or physically simulated artifacts and in some cases, a combination of both. There are a number of theoretical as well as technological considerations that needs to be addressed in order to conduct such artificial evolution. Theoretical considerations include the genetic encodings of the creature, the kinds of algorithms for driving the evolutionary process and the types of controllers suitable to act as the creature's mind. Here, the term mind is simply used to reflect the creature's artificial neural network (ANN) that act as its body's controller. We will use the terms mind and controller interchangeably throughout this thesis. As Franklin (1995) points out,

> "The overriding task of Mind is to produce the next action. Minds are the control structures of autonomous agents." (p.412)

Technological considerations for virtually simulated creatures include the physics engine for simulating the creature and its surroundings. For real-life physical robots, the technological issues include robotic platforms suitable for evolutionary design approaches and techniques for automatically generating robots with variable controllers as well as morphologies.

This thesis is about the evolution of morphology and mind in virtual organisms. We are interested in understanding how the evolution of the creature's mind affects the evolution of its behavior as well as its morphology. Although our study focuses on the evolution of simulated creatures, we believe that the results from our investigation will also help to further the understanding and development of real-life autonomous robots. On a more general level, it will also provide some useful insights into the relationship between the co-evolution and co-adaptation of body and mind in real creatures.

1.2 Motivation

There have been numerous significant contributions to this area of research over the last decade (Sims 1994a; Sims 1994b; Harvey, Husbands, Cliff, Thompson,

and Jakobi 1997) and especially in the early part of this new millennium (Nolfi and Floreano 2000; Komosinski and Rotaru-Varga 2001; Taylor and Massey 2001; Bongard and Pfeifer 2002; Hornby and Pollack 2002). We have seen how artificial evolution allows the engineering of robotic lifeforms to be fully autonomous from the initial design of morphologies and controllers to fabrication of real working robots (Lipson and Pollack 2000). Captivating communities of evolving 3D virtual organisms that live and die in a complex physics-based virtual world have given insights into the emergence of complex dynamical life-like systems (Komosinski and Rotaru-Varga 2001) as well as the general question of evolvability (Komosinski and Rotaru-Varga 2000). Studies have also shown how learning complements evolution in generating adaptive and robust controllers for wheeled robots (Floreano and Urzelai 1998; Nolfi and Floreano 1999). This is just a small sampling of the recent exciting and highly significant advancements that have been contributed by research in embodied and situated artificial creatures. The potential future contributions to the engineering, biological and cognitive sciences stemming from further research in this area are clearly evident.

The emphasis of most studies in evolving embodied artificial creatures have been on the role of genetic encodings and how different types of genotype-phenotype representations allow for greater evolvability (Bongard and Pfeifer 2001; Hornby and Pollack 2001a; Komosinski and Rotaru-Varga 2001; Bongard 2002b; Hornby and Pollack 2002). There have also been some investigations into the role of fitness functions and how they affect the direction of the evolutionary process (Floreano and Urzelai 2000; Komosinski and Rotaru-Varga 2000; Ray 2000). A very recent investigation explored how morphological complexity itself affects the emergence of more complex behavior in artificial creatures (Bongard and Pfeifer 2002). However, considerably little has been said about the role of controllers in the artificial evolution of such creatures.

In Nolfi's (2002) very recent overview of the current state-of-the-art, this gap in the literature is further supported by his remark that

"...the potential to design systems that exploit sensory-motor coordi-

nation remains largely unexplored." (p.31)

As such, there is currently a lack of understanding of how the evolution of controllers affects the evolution of morphologies and behaviors in embodied and situated creatures. It remains unclear what properties of an artificial creature's mind allow it to exhibit the desired action and form. Our motivation for this thesis stems from the fact that a better fundamental understanding of the controller's role in terms of its search space characterization, evolutionary dynamics, operational dynamics, complexity and representational power should pave the way towards our understanding of the emergence of more complex artificial creatures with a variety of morphologies and behaviors.

1.3 Research Question and Hypothesis

Life, as we all know too well, seldom allows us to survive by solely focusing on a single objective alone. Rather, it presents us with a myriad of choices and often forces us to choose between conflicting goals that in one way or another affects our chances for survival. As such, we believe that the introduction of multi-objectivity for the evolution of embodied artificial creatures will allow for this important aspect of biological life to be captured and modelled naturally as part of the evolutionary process in artificial life systems.

In this thesis, we wish to specifically answer the following research question:

Is a Pareto evolutionary multi-objective optimization (EMO)[1] approach beneficial for evolving artificial creature controllers?

Our hypothesis is that a Pareto EMO approach will reduce the computational cost of evolving effective locomotion controllers compared to non-Pareto EMO algorithms

[1] It should be noted that another acronym commonly used to refer to EMO algorithms is MOEA, which stands for Multi-Objective Evolutionary Algorithms (Deb 2001; Coello Coello, Van Veldhuizen, and Lamont 2002). In this thesis, we use only the acronym EMO to refer to this class of algorithms.

and single-objective evolutionary algorithms (EAs). Hence our main research objective is to prove or disprove that a Pareto EMO approach is advantageous for evolving locomotion controllers of artificial creatures. This objective will be achieved by comparing the trade-off between the optimized controllers and computational cost involved in finding these optimized controllers from using a Pareto EMO methodology against weighted sum EMO and single-objective EA approaches. Although the results of this thesis can be generalized to the area of evolutionary robotics, our focus will be only for simulated legged artificial organisms.

In order to answer the main research question, a number of other related sub-questions will need to be investigated as well:

1. **What are the characteristics of the underlying search space associated with finding effective locomotion controllers for artificial creatures?**

   The underlying search space associated with generating artificial creature controllers for legged locomotion needs to be understood in order to gain an idea of the difficulty associated with this problem. If the fitness landscape is highly smooth and unimodal, then a gradient-based algorithm such as greedy hill-climbing would perform well. On the other hand, if the fitness landscape is highly rugged, an evolutionary optimization approach will perhaps be of benefit in solving this problem.

2. **What types of controller architecture are suitable for evolving locomotion controllers?**

   There are a number of different ANN architectures that can be used as the artificial creature's controller. The question here is what types of ANN architecture are easier to search for generating locomotion abilities in artificial creatures. Perhaps simple feed-forward ANNs are sufficient for generating the required locomotion controllers. On the other hand, recurrent architectures might be more efficient in capturing the state-dependent dynamics of the artificial creature's legged limb motions.

3. **What is the minimum hidden layer size required to produce locomotion controllers?**

   The capacity of an ANN is determined by its so-called Vapnik-Chervonenkis (VC) dimension (Vapnik and Chervonenkis 1971), which in turn is determined by the number of free parameters in the network such as the connection weights (Haykin 1999). One way to control the weights is by controlling the number of hidden units present in the ANN. Hence, the importance of implementing a suitably-sized hidden layer within the ANN architecture needs to be ascertained. Firstly, finding the ANN controller with the minimum network size will reduce the amount of computation that needs to be carried out by the artificial creature's controller, thereby further enhancing its efficiency during operation. Secondly, to be able to use the controller as some type of complexity measure (see next item), we need to ensure that the amount of redundancy in the network is minimized as far as possible in order to avoid false indications given by large redundant networks. Thirdly, although redundancy may be beneficial for life-long learning, we need to avoid evolving networks with unseen redundancy to be able to reduce the risk of unpredictable behavior. Redundancy can be later added manually, with its corresponding effects analyzed by the designer. Thus, minimizing the number of redundant hidden units can reduce the amount of "*surprise*" (Ronald and Sipper 2001) arising from the use of biologically-inspired solutions (see Section 2.3).

4. **How can we compare between the complexities of artificially evolved creatures?**

   Another question that needs to be addressed is how can the complexity of the artificial creatures that have been evolved be measured or characterized. It is important to be able to make objective comparisons between the evolved characteristics of the artificial creatures such as their controllers, morphologies and emergent behaviors. For example is controlling a four-legged robot more complex than controlling a six-legged robot? Conversely, is a six-legged robot able to achieve more complex behaviors that are not achievable by a four-

legged robot? Being able to answer such questions will establish a methodology that allows for a better understanding of the cost trade-offs between different designs, controller requirements and operational capabilities.

5. **What effects does the simultaneous evolution of morphology together with controller have on the evolutionary optimization process?**

   The evolutionary search process for automatically generating locomotion controllers for artificial creatures can be carried out by either using a fixed hand-designed morphology for the artificial creature or by allowing self-organization to occur simultaneously for both the controller and the morphology through a co-evolutionary process, which may help to produce innovative designs. Perhaps by allowing the artificial creature's morphology to freely change and evolve as the corresponding controller evolves may ease the search space difficulty of this problem. Using co-evolution, the evolutionary search process can experiment with unconventional and previously unexplored morphological designs that may be easier to control for legged locomotion. On the other hand, adding more parameters to the evolutionary optimization process may cause an explosion in the search space subsequently causing the search algorithm to perform dismally. Hence, it is important to determine whether a co-evolutionary process is actually beneficial or otherwise in evolving locomotion controllers for artificial creatures.

1.4 Organization of the Thesis

This thesis has nine chapters and is organized as follows:

In Chapter 1, an introduction to the thesis is presented. It first provides an overview of the research field, followed by the motivation and research questions raised in the thesis. An outline of the thesis is then given and the chapter closes with a list of scientific contributions stemming from this research work.

In Chapter 2, a survey of the literature is undertaken for research conducted

in evolving artificial creatures. This review is divided into three main sections, first emphasizing the relevance of conducting such research based on the principles of embodiment and situatedness, then a survey of the work carried out using real physical robots and finally a survey of the work carried out using simulations. The survey includes both evolution of controllers alone as well as co-evolution of morphology and mind.

In Chapter 3, the virtual environment in which the experiments in this thesis are carried out is explained. Firstly, a description of the physics engine used to simulate the creature and its world is given, followed by the physical setup of the creature's morphology. Then, an explanation of the ANNs used to control the creature's movement is presented, followed by a discussion of the genotype representation of the ANN controller. Finally, the basic evolutionary and simulation parameters used in the experiments are outlined.

In Chapter 4, we investigate the question of search space difficulty associated with four different types of ANN architecture. A basic characterization of the fitness landscape involved in searching for ANN controllers that exhibit good locomotion capabilities is performed using random search, hill-climbing and random walk algorithms.

In Chapter 5, we explore the possibility of using a Pareto EMO methodology for evolving artificial creature controllers. A self-adaptive Pareto EMO algorithm called Self-adaptive Pareto Artificial Neural Network (SPANN) is presented and used to evolve ANN controllers for the artificial creature. Detailed analysis is then conducted on the evolutionary search process and comparisons made against the controllers obtained from random search, hill-climbing and random walk algorithms. The operational dynamics of the best evolved controllers are also analyzed.

In Chapter 6, we answer the main research question of whether the Pareto EMO methodology is actually beneficial for the evolution of locomotion controllers. The SPANN algorithm is compared against more conventional EAs, namely a hand-tuned EMO algorithm, a weighted sum EMO algorithm, a single-objective EA, and a recent Pareto EMO algorithm (Non-dominated Sorting Genetic Algorithm II

(NSGA-II) (Deb, Agrawal, Pratab, and Meyarivan 2000)), to verify that the self-adaptive Pareto EMO algorithm is actually beneficial for evolving artificial creature controllers. An analysis into the redundancies of the evolved networks is also given.

In Chapter 7, we tackle the question of how to compare creature complexities. A multi-objective view is presented for characterizing and comparing between the complexities of the different evolved controllers using the EMO approach. Examples are also given as to how this multi-objective approach towards understanding complexity can be useful in other disciplines.

In Chapter 8, we conduct the simultaneous evolution of both morphology and controller. The constraint of fixing the artificial creature's morphology when conducting the evolutionary optimization process is relaxed and the SPANN algorithm is augmented to enable this co-evolution of the creature's morphology and mind to occur.

In Chapter 9, the main findings from this thesis are summarized. The chapter concludes the thesis with a discussion of possible future research directions.

This thesis has an accompanying CD-ROM which can be found on the inside back cover of the thesis. The CD-ROM is divided into two main sections containing the video clips of the artificial creatures in simulation and the graphs generated during the analysis of the experimental data. The nature of this investigation necessitated the generation of a large number of graphs, not all of which could be included into the pages of the thesis itself but have been inserted into the CD-ROM. An index to the contents of the CD-ROM can be found in Appendix A.

1.5 Original Contributions

A list of original scientific contributions arising from this thesis is given in this section.

- A Pareto evolutionary approach for evolving locomotion of a quadruped is presented. Although Pareto methods have been used for designing intelligent control systems (Tan and Li 1997; Gacogne 1997; Coello Coello, Christiansen,

and Aguirre 1998; Pirjanian 1998), no study to our knowledge has attempted to use such methods for evolving locomotion of a quadruped. Previous methods for conducting this type of evolution have focused on using single-objective EAs or weighted sum EMO methodologies for optimizing the desired behavior. The Pareto EMO approach is shown to offer significant advantages for evolving artificial creature controllers. This will open up an entirely new paradigm into evolving controllers not only for simulated quadrupeds but also for all other types of artificial life and physical robots.

- An analysis of the fitness landscape for quadruped locomotion and a critical evaluation of the current literature for fitness landscape analysis are presented (Chapter 4). The analysis provides an insight into the variety of fitness landscape features that can significantly affect the outcome of evolutionary searches for locomotion controllers. An understanding of the underlying search space characteristics is paramount towards the design of more effective search strategies and optimization algorithms for the purpose of generating quadruped locomotion controllers. The deficiencies noted with current methods of characterizing fitness landscapes will pave the way for more insightful and practical solutions to be devised for future investigations into the search spaces of artificial creature evolution.

- A systematic study of the relationship between quadruped locomotion and controller size is presented, and a modified version of the SPANN algorithm suitable for evolving ANNs for robotic control is developed (Chapter 5). A fundamental understanding of the size requirement of ANN controllers for quadruped locomotion will allow for more efficient use of computational resources in the control of autonomous robot and virtual creature locomotion. The proposed SPANN algorithm allows for the use of a Pareto approach for the automatic generation of ANNs that can serve as effective autonomous control units for virtual artificial creatures.

- Presenting the advantages of SPANN by comparing it with a hand-tuned,

weighted sum, single-objective, and NSGA-II algorithms (Chapter 6). The much lower computational cost offered through the self-adaptive Pareto EMO approach can significantly reduce the amount of time required to carry out artificial life and evolutionary robotics experiments for finding effective artificial creature controllers.

- First attempt to formulate the problem of hierarchical complexity as a multi-objective optimization problem and establish EMO as a platform for studying complexity (Chapter 7). The measurement and comparison of complexity between different objects have always been a problematic issue across multiple research disciplines. An entirely different perspective towards characterization of hierarchical complexity can be achieved by taking a multi-objective viewpoint. The multi-objective characterization of complexity can open up radically different avenues into complexity research and in general how researchers think about complexity. We show that complexity characterization can be carried out in a simple and practical manner using an EMO approach. This highly accessible method of capturing complexity has significant implications not only within the scope of artificial life and evolutionary robotics but across a much wider spectrum of research fields.

- Proposing a methodology for studying the impact of morphological constraints on behavior and evolution (Chapter 8). The imposition of pre-designed morphologies on both physical robots and simulated agents may require more complex controller requirements as well as entailing a more involved evolutionary search. The co-evolution of both morphology and controller through the relaxation of certain morphological constraints can lead to similarly good locomotion behavior as well as new and interesting artificially evolved morphological designs. This can allow for previously unexplored robot bodies or simulated characters to be engineered and synthesized. At the same time, the co-evolutionary approach represents an important step towards truly evolvable materials and physical constructs, especially in the field of nanotechnology.

Chapter 2

Evolving Artificial Creatures

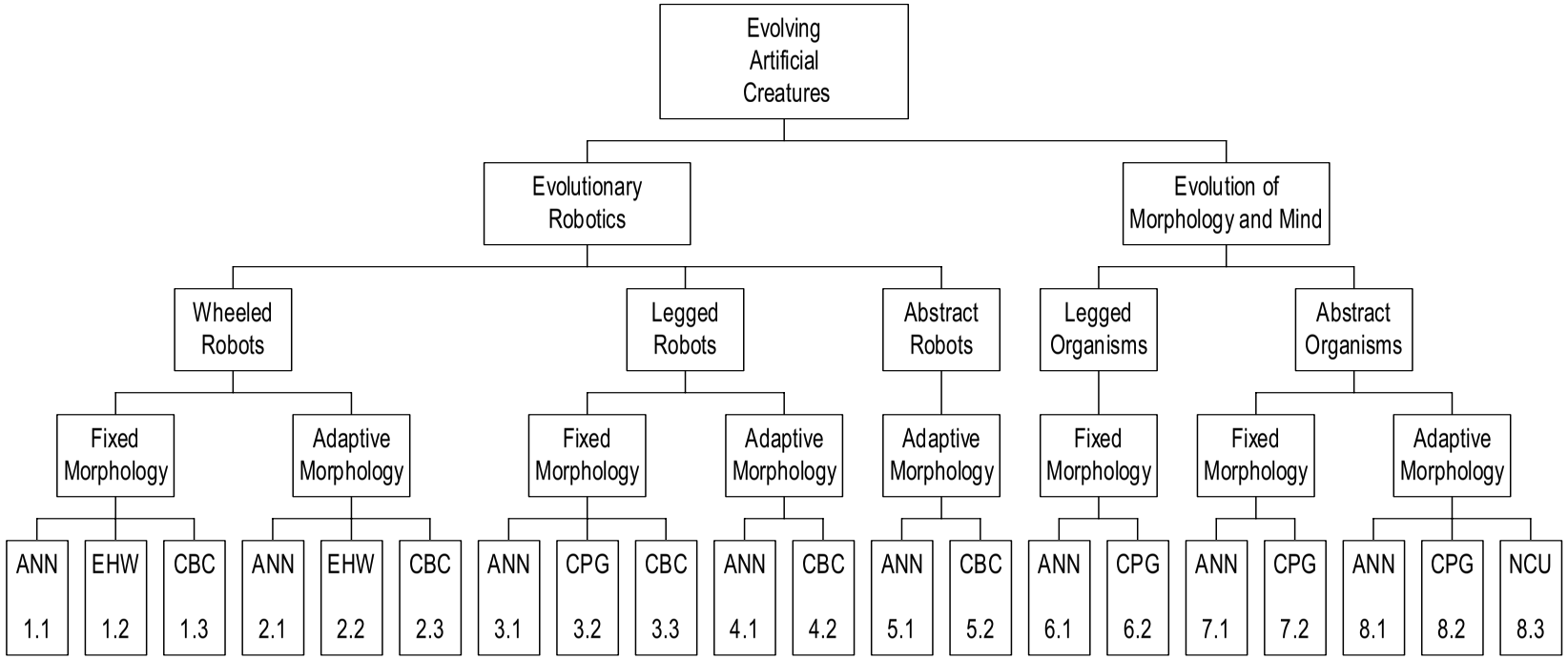
2.1 Introduction

As we have seen from the brief introduction given in Chapter 1, the artificial evolution of embodied and situated creatures can be classified into two groups: (1) the evolution of virtual creatures in simulation, and (2) the evolution of real physical robots. Over the last decade, work on evolving robots has become a mainstream effort in robotics and the field has come to be known as evolutionary robotics (Nolfi and Floreano 2000). On the other hand, the evolution of virtual abstract creatures in simulation has not reached the level of maturity achieved by its physical counterpart. As such, there is no commonly agreed upon term that refers to this latter type of work. Some of the keywords used to describe the evolution of virtual abstract creatures in simulation include virtual embodied evolution (Bongard and Paul 2000), virtual creature evolution (Komosinski and Rotaru-Varga 2001), body-brain co-evolution (Hornby and Pollack 2002) and evolution of morphology and behavior (Taylor 2002). In this thesis, we will use the term evolution of morphology and mind to refer to this class of work. As explained earlier in Section 1.1, the word mind here is used to refer to the ANN that acts as the artificial creature's controller.

It should be noted however that there is no strict delineation between the two fields of physical and simulated evolution of artificial creatures. As we will see later in this chapter, a significant proportion of the work in evolutionary robotics

does actually involve simulation of real robots to reduce the steep time requirements when conducting evolution on real physical robots (Mataric and Cliff 1996). A series of studies on "minimal simulations" has shown that if the simulation faithfully captures the robot's operation in its environment including the presence of noise in sensors and motors, then evolved controllers in simulation can be successfully transferred to real world robots (Jakobi, Husbands, and Harvey 1995; Jakobi 1997b; Jakobi 1997a; Jakobi 1998). Additionally, some of the highly abstract creatures evolved in simulation, which are far from the design or workings of any real-life robots, have actually been literally "fleshed out" to become tangible, physical manifestations of the real world (Lipson and Pollack 2000; Hornby, Lipson, and Pollack 2001).

This chapter begins with an overview of the importance of embodied and situated evolution. The relevant literature concerning the evolution of real physical robots is then reviewed, followed by a review of the evolution of morphology and mind in simulated artifacts. Figure 2.1 in conjunction with Table 2.1 provides a road-map to the literature surveyed in this chapter on the evolution of different types of artificial creatures. The final level of categorization in Figure 2.1 have been assigned numerical tags of which the corresponding entries in Table 2.1 list the details of the research work in that grouping.



Key to Abbreviations Used for Controllers:

ANN - Artificial Neural Network

EHW - Evolvable Hardware

CBC - Command-Based Controller

NCU - No Controller Used

CPG - Central Pattern Generator

Figure 2.1: Road-map to research on evolving artificial creatures. Numerical tags correspond to entries in Table 2.1.

Ref.	Year	Authors
1.1	1996	Eggenberger
	1997	Lund and Hallam
	1998	Floreano and Mondada
		Floreano and Urzelai
		Husbands, Smith, Jakobi, and O'Shea
	1999	Nolfi and Floreano
	2000	Floreano and Urzelai
	2001	Floreano and Mattiussi
		Floreano, Nolfi, and Mondada
		Floreano and Urzelai
		Hulse, Lara, Pasemann, and Steinmetz
		Husbands, Philippides, Smith, and O'Shea
		Pasemann, Steinmetz, Hulse, and Lara (a)
		Pasemann, Steinmetz, Hulse, and Lara (b)
		Lara, Hulse, and Pasemann
		Smith, Husbands, and O'Shea (a)
		Smith, Husbands, and O'Shea (b)
	2002	Eggenberger, Gomez, and Pfeifer
		Floreano, Schoeni, Caprari, and Blynel
		Nolfi
		Philippides, Husbands, Smith, and O'Shea
		Smith, Husbands, Philippides, and O'Shea
		Smith, Philippides, Husbands, and O'Shea
		Watson, Ficici, and Pollack
1.2	1995	Thompson
	1996	Keymeulen, Durantez, Konaka, Kuniyoshi, and Higuchi
	1997	Thompson
	1998	Keymeulen, Iwata, Konaka, Suzuki, Kuniyoshi, and Higuchi
1.3	1996	Nordin and Banzhaf
2.1	1997	Harvey, Husbands, Cliff, Thompson, and Jakobi
2.2	1997	Lund, Hallam, and Lee
2.3	1996	Lee, Hallam, and Lund
3.1	1992	Beer and Gallagher
	1996	Gallagher, Beer, Espenschied, and Quinn
	1997	Gruau
	1998	Jakobi
		Kodjabachian and Meyer (a)
		Kodjabachian and Meyer (b)
	1999	Reeve
	2001	Fujii, Ishiguro, Aoki, and Eggenberger
		Otsu, Ishiguro, Fujii, Aoki, and Eggenberger
		Paul and Bongard
3.2	2001	Reil and Massey

Ref.	Year	Authors
	2002	Reil and Husbands
3.3	1998	Gomi and Ide
	1999	Hornby, Fujita, Takamura, Yamamoto, and Hanagata
	2000	Hornby, Takamura, Yokono, Hanagata, Yamamoto, and Fujita
4.1	2001	Bongard and Paul
4.2	1997	Arnold
5.1	1993	Cliff, Harvey, and Husbands
	1994	Dellaert and Beer
	1994	Harvey, Husbands, and Cliff
	1996	Cliff and Miller
	1997	Husbands, Harvey, Jakobi, Thompson, and Cliff
	1999	Lichtensteiger and Eggenberger
	2000	Lipson and Pollack
5.2	1999	Dittrich, Skusa, Banzhaf, and Kantschik
6.1	1998	Ijspeert, Hallam, and Willshaw
	1999	Ijspeert
		Ijspeert, Hallam, and Willshaw
		Ijspeert and Kodjabachian
	2001	Ijspeert
	2002	Bongard (a)
	2002	Bongard and Pfeifer
6.2	2000	Ijspeert
		Ijspeert and Arbib
7.1	2002	Mandik
7.2	1997	Gritz and Hahn
8.1	1994	Sims (a)
		Sims (b)
	1999	Komosinski and Ulatowski
	2000	Bongard and Paul
		Komosinski
		Ray
	2001	Bongard and Pfeifer
		Hornby and Pollack (a)
		Komosinski, Koczyk, and Kubiak
		Komosinski and Kubiak
		Komosinski and Rotaru-Varga
		Taylor and Massey
	2002	Bongard (b)
		Hornby and Pollack
8.2	2001	Hornby, Lipson, and Pollack
		Hornby and Pollack (b)
8.3	1997	Eggenberger

Table 2.1: Summary of literature survey on evolution of artificial creatures. Num-

bered references refer to the numerical tags assigned in Figure 2.1.

2.2 Situatedness and Embodiment

The importance of embedding the study of evolving artificial creatures within the twin principles of situatedness and embodiment is perhaps best exemplified by how natural evolution occurs in real biological organisms. Natural creatures such as animals and insects have bodies and are situated in a physical environment. Their skills and behaviors are developed autonomously through the intimate interplay with their environment. As such, in order to create artificial creatures that might possess some of these novel properties exhibited by real creatures, such systems must be built based on the principles of situatedness and embodiment.

This view of intelligence as an emergent phenomenon of embodied and situated artifacts is regarded by many researchers to be the foundation for successful design and implementation of artificial agents. On embodiment, Varela (1995) pointed out that

> "Cognition depends on the kinds of experience that come from having a body with various sensorimotor capacities." (p.15),

while Brooks (1995) stressed that

> "The robots have bodies and experience the world directly — their actions are part of a dynamic with the world and have immediate feedback on their own sensations." (p.29),

and Arkin (1998) stated that

> "A robot has a physical presence (a body). This spatial reality has consequences in its dynamic interactions with the world . . . " (p.26).

Following on to situatedness, Varela (1995) explained that

> "The individual sensorimotor capacities are themselves embedded in a more encompassing biological and cultural context." (p.15),

while Brooks (1995) highlighted that

> "...robots are situated in the world — they do not deal with abstract descriptions, but with the here and now of the world directly influencing the behavior of the system." (p.29),

and Arkin (1998) stated that

> "The robot is an entity situated and surrounded by the real world. It does not operate upon abstract representations of reality, but rather reality itself." (p.26).

In Dautenhahn's work with socially intelligent robots (Dautenhahn 1996; Dautenhahn 1999), the importance of embodiment was discussed at length in designing reactive cognitive architectures in physical robots and other artificially intelligent agents that can exist in simulation. Dautenhahn (1996) highlighted the fact that

> "...there is much evidence to support the assumption that cognitive capabilities are only possible through the interaction of body and mind, i.e. that the body is not simply used by the mind, but that there is a co-development and mutual shaping of cognitive abilities on the one hand and bodily skills and experiences on the other hand. The body is not a fixed and pregiven 'actuator device', but it is a dynamic and ontogenetically evolving entity." (p.27).

Furthermore, Dautenhahn (1996) argued that the study of embodied and situated artifacts will play a significant role in bridging the gap between phenomenological understanding and the computation-theoretic approaches normally adopted in cognitive science, artificial intelligence and artificial life studies.

Mataric (1997) addressed the issue of how physical embodiment is related to cognition and reviewed both biological and artificial studies that have endeavored to answer this question. It was argued that artificial systems are preferable over biological systems as although biological data are abundant, they are often disconnected and incomplete due to the restrictions that apply when working with real

rather than artificial life. On the other hand, the use of artificial systems allows the researcher complete freedom to experiment with whatever aspects of embodiment and cognition that are of interest, in particular ablation of neural pathways, amputation of limbs and/or other forms of disablement of sensory-motor capabilities, which are central to the question of the role of embodiment in higher-level cognition. In the author's own study, artificial mobile agents were used to answer a number of key questions relating to social group behavior as well as imitative behavior, and how these behaviors are in turn related to embodiment and cognition.

Nolfi and Floreano (2002) importantly pointed out that for an external observer, designing such situated and embodied creatures capable of autonomously developing the desired behavior through dynamical interactions with their environments is a very complex task. They further explained that there were two ways in which this can be achieved: (1) by painstakingly recreating the artificial creature through careful mimicking of natural organisms, or (2) by employing an artificial evolutionary process that allows for self-organization to occur automatically. As such, this makes the evolution of embodied and situated creatures a prime candidate for evolutionary computation techniques.

2.3 Evolutionary Robotics

Evolutionary robotics is defined to be the synthesis of autonomous robots using artificial evolutionary methods (Nolfi and Floreano 2000). An early review of this field of research is given by Mataric and Cliff (1996) where the majority of studies focused mainly on the evolution of control structures only. A more recent overview highlights the move of evolutionary robotics into evolving both the control and morphology of robots where the interplay between brain and body is considered to be a crucial factor in the successful synthesis of autonomous robots (Nolfi and Floreano 2002). A thorough treatment of the field can be found in the seminal textbook written by Nolfi and Floreano (2000) on this subject.

As pointed out by Harvey (1997), the design of controllers for robots is

a complex task not suited to human divide-and-conquer design strategies. There are 3 major problems: (1) it is not obvious how the controller system should be decomposed, (2) interactions are not limited to direct connecting links but are also mediated through the environment, and (3) interactions between sub-parts grows exponentially as system complexity increases. Thus, evolutionary approaches to controller design are desirable, where the only benchmark is the overall behavior that should be achieved by the system.

However, Ronald and Sipper (2001) recently pointed out that emergence stemming from the use of biologically-inspired solutions in engineering problems may be problematic because unexpected and sometimes unwanted results or behaviors might arise. Using the so-called emergence test, it was claimed that evolutionary robotics exhibited mild emergence where the degree of surprise is limited to well-defined boundaries *(unsurprising surprise)*. On the other hand, traditional hard-wired engineering solutions exhibited no surprise *(unsurprising)* while artificial life exhibits a very high degree of surprise *(surprising surprise)*. Nonetheless, it was surmised that emergence in engineering solutions that draw on inspirations from nature such as evolutionary robotics and the related reliability issues are unavoidable consequences if the desire is to design smart, adaptive and evolvable machines. In general, evolutionary robotics can be grouped into three main categories, those involving the evolution of (1) wheeled, (2) legged, and (3) abstract robots.

2.3.1 Wheeled Robots

A hybrid genetic programming (GP)/genetic algorithm (GA) methodology was used to evolve both the controller and parameters of a wheeled robot's morphology in simulation (Lee, Hallam, and Lund 1996). The controller consisting of a tree-like program was evolved using the GP part of the system while morphological parameters such as the robot's body size, wheel radius and wheel base size encoded in a linear string of real numbers were evolved using the GA part of the system. Individuals were assessed for obstacle avoidance behaviors using a fitness function that combined multiple terms such as distance from obstacles, forward speed and

rotating speed into a single objective. It was claimed to be the first study which co-evolved both the controller and morphology of robots and concluded that because the evolved controller only functioned within the co-evolved body, the evolution of the body component played a significant role in the success of the evolutionary process. An island-GA model was used to maintain genetic diversity during the evolutionary process. In a related study using simulations, Khepera wheeled robots were shown to require only simple perceptron controllers that directly connected sensors to motors for evolving behaviors such as exploration and homing (Lund and Hallam 1997). It was claimed that the robot's perception of its environment's geometries allowed time-related components to be encoded without requiring any recurrent connections in the controller. GP alone has also been used to evolve controllers for Khepera robots for obstacle avoidance and object tracking behaviors utilizing a combination of simulated and real-world testing of evolved controllers (Nordin and Banzhaf 1996).

The Species Adaptation Genetic Algorithm (SAGA) algorithm was used to evolve both the controller and visual morphology parameters for simple navigational tasks in a two-wheeled mobile autonomous robot (Harvey, Husbands, Cliff, Thompson, and Jakobi 1997). The desired behavior was evolved within 50–100 generations using 40–60 individuals that were evaluated using a simple single-objective distance-based fitness function. SAGA allows for increases in length to genotypes and hence it was argued that it permitted incremental evolution to occur during the evolutionary process. Conversely, Eggenberger (1996) reported the use of biological cell differentiation techniques in order to reduce the length of the genotype encoding when evolving neural network controllers for Khepera robots in simulation. It was claimed that using such a developmental method, the genome need not necessarily increase in length whenever the number of neurons increased since no specific data relating to the presence or otherwise of neurons need to be stored in the genome, which will now be specified as part of the cell differentiation process rather than being directly encoded for in the genome. This cell differentiation system has subsequently been used to evolve only the morphologies of static 3D virtual organisms

(Eggenberger 1997) and more recently to grow the connectivity of a neural network for controlling a foveating retina of a real physical robot (Eggenberger, Gomez, and Pfeifer 2002).

Related work with wheeled robots have also shown promising results in robustness and the ability to cope with changing environments by evolving plastic individuals that are able to adapt both through evolution and lifetime learning (Floreano and Mondada 1998; Floreano and Urzelai 1998; Nolfi and Floreano 1999; Floreano and Urzelai 2000; Floreano and Urzelai 2001). A number of different reactive navigation behaviors were generated using evaluation functions that typically included different terms for rewarding speed, wall avoidance and straight-line motion combined into a single objective. Instead of evolving the synaptic weights, the learning rules governing the behavior of individual synapses were evolved when generating a neural network controller for Khepera robots. It was demonstrated that the evolved controllers were adaptive to changes in the environment due to their synaptic plasticity. Lifetime learning or ontogenetic adaptation has several adaptive functions within evolution: (1) allowing for individuals to adapt to fast-changing environmental conditions, (2) channelling information extracted from the environment to evolution, (3) helping to guide evolution, (4) reducing genotype length, and (5) maintaining genetic diversity (Nolfi and Floreano 1999). Learning and evolution were shown to be able to solve tasks that evolution alone could not solve. Performance increases were also noticed even when the learning tasks differed from the selection tasks. Learning individuals were thus better adapted to changing environments than non-learning individuals. Interaction between learning and evolution deeply altered both these processes in that learning enabled evolution to extract supervision information from the environment. In terms of generality, plastic-general individuals required less complex control systems compared to full-general individuals. Ontogenic adaptation has also been studied in a competitive co-evolutionary context of predator-prey simulations using Khepera robots (Floreano, Nolfi, and Mondada 2001).

Pure reactive agents that do not use any internal representation were shown

to able to solve complex tasks through the use of sensory-motor coordination only (Nolfi 2002). By exploiting agent-environment interactions, these embodied artificial creatures were able to coordinate perception and action that enabled them to perform complex tasks without needing to react differently to the same sensory states in different contexts. The experiments involving physical agents were carried out using Khepera robots and neural networks weights were evolved for the control of the agents. Sensory-motor coordination allowed the robots to (1) select the most effective feedback, (2) simplify harder tasks, (3) exploit emergent behaviors, and (4) exploit environmental constraints. Pure reactive agents although effective were found to be sub-optimal in most conditions. As a remedy, it was suggested that more complex behaviors could be allowed to emerge through a simple process of adding internal representations to the existing reactive behaviors.

In a departure from classical connectionist models, Floreano, Schoeni, Caprari, and Blynel (2002) recently demonstrated the use of evolutionary spiking neurons for the control of an autonomous microbot. A single "spike" in a spiking neural network is a discrete binary event that simply encodes whether a stimulus is present or absent. Instead of using conventional non-linear, real-valued sigmoidal activation functions, the use of spiking neurons in neural circuits were shown to transfer easily to microcontrollers by virtue of their binary nature, which can be mapped onto low-level digital circuits using only a few logic operations such as AND and NOT. In an earlier study, it was shown that viable controllers were easier to evolve using spiking neurons than sigmoidal neurons for a vision-based navigation task of a Khepera robot (Floreano and Mattiussi 2001).

Comparatively small neural networks that utilized recurrent connections were shown to be capable of producing good obstacle avoidance and light-seeking behaviors in Khepera robots (Pasemann, Steinmetz, Hulse, and Lara 2001a; Pasemann, Steinmetz, Hulse, and Lara 2001b) using the ENS^3 (Evolution of Neural Systems by Stochastic Synthesis) algorithm. A weighted sum of different speed and navigation objectives were combined into a single-objective function for the evaluation of evolved networks. The simplest evolved networks did not use any hidden

units and it was also demonstrated that larger networks were not necessarily more robust than smaller ones. In a related study, separately evolved neuromodules for obstacle avoidance and light-seeking behaviors were combined together to produce a single controller with both behaviors (Lara, Hulse, and Pasemann 2001) by evolving additional interface neurons and synapses for the interconnection between these two neuromodules. It was also shown in another related experiment that the evolved controllers were robust and performed well in both simulated and actual robots (Hulse, Lara, Pasemann, and Steinmetz 2001).

The control structures consisting of ANNs for a population of robots were evolved using a fully decentralized EA (Watson, Ficici, and Pollack 2002). The EE (Embodied Evolution) methodology was defined as conducting evolution in a group of real physical robots where evaluation, selection, and reproduction took place by and between robots in a distributed, asynchronous and autonomous manner. The robots were simple two-wheeled self-designed mobile agents with inter-agent communication capabilities. Evolved controllers outperformed hand-designed controllers for a phototaxis task.

A gaseous signalling mechanism was used in the GasNet algorithm for generating robot controllers in visual discrimination and navigation tasks (Husbands, Smith, Jakobi, and O'Shea 1998; Husbands, Philippides, Smith, and O'Shea 2001). The fitness of generated controllers was evaluated using a single function that combined the weighted sum of navigational scores. Although the neural networks using the gaseous signalling mechanisms could be evolved in fewer generations compared to neural networks that did not use these mechanisms, implying a less difficult search space in the former neural networks, all the standard random sampling measures used to discriminate between the two different search spaces failed to show any discernable differences between these evolutionary systems (Smith, Husbands, and O'Shea 2001b). Further analysis showed that the evolutionary robotics search space exhibited phases of neutral evolution (Smith, Husbands, and O'Shea 2001a). The population as a whole was shown to move significantly in the genotype space during such phases of neutrality and was not trapped at a local optimum in the fitness

landscape. However, no evidence could be found to indicate that neutral adaptation acted as a scaffolding for later transitions to higher fitness levels. As such, it was concluded that neutrality did not play any useful role in this particular evolutionary robotics search space.

It was later shown that the combined effects of increased neutrality and decreased ruggedness in evolutionary robotics search spaces allowed for greater evolvability (Smith, Philippides, Husbands, and O'Shea 2002). It was argued that phenotypic stability and genetic instability were prerequisites if successful evolution were to occur in an organism. Four different GasNet neural network models acting as controllers for simulated mobile robots in a shape discrimination task were implemented with varying degrees of redundancy and coupling to elucidate these effects. More recently, Smith, Husbands, Philippides, and O'Shea (2002) showed that the high success rates of GasNets neural networks in the visual discrimination task was due to temporal adaptivity and argued that this property is fundamental for the generation of adaptive behavior. Recent related work has also extended the family of GasNet neural networks to include more details of biological gaseous signalling mechanisms into two new versions called the *plexus* and *receptor* models, which were shown to be more evolvable than the earlier version of GasNet (Philippides, Husbands, Smith, and O'Shea 2002).

2.3.2 Legged Robots

The pioneering work of Beer and Gallagher (1992) documented the use of GA to evolve continuous-time recurrent neural networks for controlling the legged locomotion of a hexapod insect, although this study was conducted using a highly simplified physics model. It was shown in a later study that the evolved controllers could still perform the locomotion successfully when transferred to a real hexapod robot (Gallagher, Beer, Espenschied, and Quinn 1996). Related studies based on this simplified six-legged hexapod model have been conducted to investigate the evolution of neural network architectures rather than synaptic weights alone using a developmental scheme specified by the Simple Geometry Oriented Cellular Encoding

(SGOCE) algorithm (Kodjabachian and Meyer 1998a; Kodjabachian and Meyer 1998b).

Arnold (1997) investigated the generation of legged locomotion for four and six-legged virtual creatures using spectral synthesis (involving Fourier transforms). The control system was algorithm-based and did not make use of any sensory input information. Evolution was used only to tune the parameters of the algorithmic controllers and limb attributes for optimizing a single-objective function of maximizing horizontal distance achieved within a designated time period. In a later study by Reeve (1999), the control mechanism based on different models of neural networks for generating legged locomotion for a range of fixed morphology robots were evolved in simulation using a simple GA. It was found that simple single-termed fitness measures based on performance attributes such as speed was sufficient to generate the desired behavior and that more complex fitness measures relating to inner workings of neurons and joints were not advantageous. It was also found that higher-order neural networks were significantly better at performing the required tasks and that very densely connected controllers performed better than sparsely connected ones.

A dynamically-rearranging neural network (DRNN) was evolved to act as a controller for legged locomotion in a simulated biped robot (Fujii, Ishiguro, Aoki, and Eggenberger 2001). Generated controllers were assigned fitness values based on a single-objective function of horizontal movement achieved. Neuromodulators were used to dynamically change synaptic weights as well as network architecture by activating and blocking neurons and synapses. However, it was observed that many of the evolved controllers did not actually make use of the modifiable synaptic weights, in other words normal neural networks with fixed synaptic weights would have sufficed. Nonetheless, it was claimed that the DRNN would have exhibited superior performance in a changing environment due to their polymorphic characteristics although this was not investigated using the biped robot. In related work using a simulated quadruped robot, a DRNN was again evolved to act as a controller for legged locomotion (Otsu, Ishiguro, Fujii, Aoki, and Eggenberger 2001). It was claimed that the controllers generated were adaptive to changes in the environment

(retardant forces and uneven slopes) due to the neuromodulations present in the DRNN. However, as no analysis was provided on the actual dynamics of the neuromodulators during the legged locomotion of the quadruped, it remains unclear what roles these elements actually played towards the generation of a successful legged locomotion in the changing environments.

Both the controller and morphology of a biped robot were evolved using a GA with a simple single-objective fitness function based on horizontal distance travelled (Paul and Bongard 2001). It was claimed that the experiments produced the first reported results of stable bipedal locomotion achieved through the optimization of both controller and morphology. An interesting point to note was that only 6 out of the 60 evolutionary runs were successful in evolving a stable gait. The architecture of the recurrent neural networks that were used as the controllers remained fixed with only the synaptic weights being evolved. Also, only certain parameters of robot's morphology were allowed to be modified during evolution. A related study using similar biped robots where both the controller and morphology were co-evolved found that the inclusion of certain morphological parameters allowed for fitter individuals to be discovered by evolutionary search (Bongard and Paul 2001). It was shown that fitter individuals did not arise simply because a better morphology was found but rather the addition of morphological parameters into the genotype space allowed for extra-dimensional bypasses to be formed in the higher dimensional search space, thereby allowing the evolutionary search to find these fitter individuals. This phenomenon facilitated the connection of otherwise isolated adaptive peaks in the objective space, making it easier for the evolutionary search process to proceed smoothly from one adaptive peak to the next.

Central pattern generators (CPGs) were evolved as controllers for generating planar walking behaviors in two different physically simulated bipeds (Reil and Massey 2001). It was shown that using the appropriate mechanical construction, Hopfield neural network controllers and optimization through a GA with a single-objective distance-based fitness function, minimal bipedal locomotion can be achieved by CPGs that do not require sensor inputs. In the second more sophis-

ticated biped, incremental evolution was used where a weak stabilizing controller was used during the initial stages of evolution and later removed after a certain fitness level was achieved. The lower portions of the more sophisticated biped's legs were implemented as passive limbs to allow for a more anthropomorphic gait to emerge. Only 10% of the first biped's evolutionary runs produced successful controllers whereas 80% of the second biped's evolutionary runs produced successful controllers. However no analysis was given on whether the two search spaces differed significantly in terms of optimization difficulty.

In a related study, CPGs were again evolved to generate bipedal locomotion in a simulated robot in a real-time physics environment (Reil and Husbands 2002). Once more, it was shown that no sensory inputs were necessary to generate successful straight-line walking behavior although this was achieved only on a homogenous planar surface. It was suggested that the fitness landscape underlying the evolutionary search space of the recurrent ANN architecture is very smooth leading to successful evolution of controllers despite using only a very simple single-objective fitness function based on a combination of two objectives of maximizing distanced travelled from origin and minimizing occurrences of falling below a certain height threshold for the robot's center of gravity. However it was also reported that only 10% of the evolutionary runs resulted in stable controllers and that an additional fitness term that rewarded cyclic activity in the ANN was necessary to improve the success rate. The authors also noted a shortfall in the experimental setup in that the effect of network size on the efficiency of the approach was not studied. A number of important contributions of the evolutionary robotics approach to designing controllers for legged locomotion of artificial creatures were highlighted: (1) fully automated process that allows for changes or additions to the creature's structure to be accommodated very easily through re-evolution, (2) diversity of solutions, and (3) relatively cheap evolutionary computational requirements.

Real physical robots have also been used to study the generation of legged locomotion using EAs. Online evolution was used by Gruau (1997) and Gomi and Ide (1998) to generate static gaits for an octopod robot, and by Hornby and his co-

researchers to generate dynamic gaits for a Sony quadruped robot (Hornby, Fujita, Takamura, Yamamoto, and Hanagata 1999) as well as for the Sony entertainment robot dog AIBO (Hornby, Takamura, Yokono, Hanagata, Yamamoto, and Fujita 2000). The cellular encoding method of (Gruau 1994) was used to evolve not only the weights but also the architecture of the neural network controller for the octopod robot and also relied on interactive user assignment of fitness values rather than integrating a fully automated fitness assignment into the artificial evolutionary process (Gruau 1997). Jakobi (1998) utilized his "minimal simulation" method to also evolve gaits in simulation for the same octopod robot in order to reduce the time requirements of evolution on the real physical robot.

2.3.3 Abstract Robots

The neural network controller and visual morphology for visually guided behaviors in a specialized gantry robot was evolved using the SAGA algorithm (Harvey 1992) for a visual discrimination task (Harvey, Husbands, and Cliff 1994; Husbands, Harvey, Jakobi, Thompson, and Cliff 1997). Minimal vision systems and small networks were found to be sufficient for generating the required behaviors using a weighted sum combination of navigational scores as the evaluation function. Small population sizes and small number of generations were also sufficient for successfully evolving these controllers. A good choice of control system primitives were suggested as the main reason for the success of these evolutionary runs. Work has also been carried where only the morphology of a compound eye on an abstract robot was evolved while the neural network controller was kept fixed (Lichtensteiger and Eggenberger 1999). Cliff, Harvey, and Husbands (1993) have also conducted work on evolving both the visual morphology and recurrent neural network controller of a mobile robot. In a later related study, Cliff and Miller (1996) also evolved both visual morphologies and neural controllers of predator-prey agents in a competitive co-evolutionary environment although the simulation was only carried out in a 2D world. Using developmental methods, the bodies and controllers of autonomous 2D agents have also been co-evolved in a study by Dellaert and Beer (1994).

GP was used to evolve controllers for a robot with manually reconfigurable morphology called a random morphology robot (RM robot) (Dittrich, Skusa, Banzhaf, and Kantschik 1999). Evaluation of controller fitness was carried out using a single-objective function consisting of weighted scores for the robot's speed and distance. The fitness landscape was found to be highly dynamic because the robot moves around on a carpeted floor and hence encounters situations with different levels of difficulties arising from the directionality of individual carpet strands. It was shown that discrimination between good and bad individuals was hard during certain periods of the evolutionary process where the noise level was high. Hence it was proposed that reference individuals be employed to enable a differential fitness value to be calculated for evolving individuals in order to better capture the actual performance of individuals throughout the highly variable evolutionary periods. Nevertheless, it was later observed that there were periods where the fitness landscape oscillated, which created a problem for the proposed relative fitness methodology as well.

Lipson and Pollack (2000) combined both simulated and physical approaches for evolving simple robots composed of bars, actuators and artificial neurons for the single objective of maximizing horizontal distance moved. The authors claimed that to fully realize artificial life, autonomy must be achieved not only at the level of power and behavior but also at the levels of design and fabrication. They demonstrated this point in their experiments where artificial evolution was conducted to automatically design abstract robots that could perform locomotion in simulation and then the best virtual designs were fabricated into real robotic body parts using 3D thermoplastic solid printing techniques. The results from testing the physical versus the virtual robots showed that in one case, the distance travelled was almost identical while in the two other cases, the distances travelled were quite dissimilar although it was argued that the overall control and mechanics of the motion were still maintained when moving from simulation to reality.

2.3.4 Evolvable Hardware

Evolvable hardware circuits in the form of field programmable gate arrays (FPGAs) were utilized to evolve obstacle avoidance controllers for Khepera robots (Thompson 1995; Thompson 1997) and was claimed to be the first example of intrinsic hardware evolution (Harvey 1997), where every actual hardware specified during the evolutionary process was tested in situ rather than in simulation. Another series of studies also utilized FPGAs as evolvable controllers for producing visual tracking and obstacle avoidance behaviors in Khepera robots (Keymeulen, Durantez, Konaka, Kuniyoshi, and Higuchi 1996; Keymeulen, Iwata, Konaka, Suzuki, Kuniyoshi, and Higuchi 1998). Solutions generated were evaluated using a fitness function that took a weighted sum combination of two objectives of minimizing the robot-target distance and minimizing the number of steps required to complete the task. It has been argued that *true evolvable hardware* should allow for both control circuits and body plans to be evolved (Lund, Hallam, and Lee 1997). Such true evolvable hardware using a modified version of the Khepera robot with a reconfigurable auditory morphology was developed by Lund, Hallam, and Lee (1997) as a framework for studying the evolution of phonotaxis in crickets although no result from actual experimentation was reported.

2.4 Evolution of Morphology and Mind

The study of evolving physically situated and virtually embodied artificial creatures has been a hotbed of research in recent years. The availability and maturation of commercial-off-the-shelf physics engines coupled with the dramatic increase of personal computing power have encouraged widespread research into this intriguing field of artificial life (Taylor and Massey 2001). Not surprisingly, there were notably few significant advancements in this field since the pioneering work of Sims (1994a, 1994b) until very recently.

Research in this area generally falls into two categories: (1) the evolution of controllers only for creatures with fixed morphologies (Gritz and Hahn 1997;

Taylor 2000; Bongard and Pfeifer 2002; Mandik 2002), and (2) the evolution of both the creatures' morphologies and controllers simultaneously (Sims 1994a; Sims 1994b; Komosinski and Ulatowski 1999; Bongard and Paul 2000; Komosinski 2000; Komosinski and Rotaru-Varga 2000; Ray 2000; Bongard and Pfeifer 2001; Hornby and Pollack 2001a; Komosinski and Rotaru-Varga 2001; Taylor and Massey 2001; Bongard 2002b; Hornby and Pollack 2002). Some work has also been carried out in evolving only the morphology alone for static 3D virtual organisms (Eggenberger 1997) and evolving morphology with a fixed controller (Lichtensteiger and Eggenberger 1999).

The idea of using artificial evolutionary methods to automatically generate 3D embodied virtual creatures was first introduced by Karl Sims in 1994. A proprietary physics-based simulation system was implemented to evolve both the morphology and neural systems of virtual creatures using a directed graph grammar. Conventional artificial evolution was used to evolve specific behaviors such as swimming, walking, jumping and light-following (Sims 1994b) and a competitive co-evolutionary method was used to evolve creatures for resource acquisition (Sims 1994a). The fitness of evolved creatures was judged using simple single-objective functions such as speed and height achieved. Although highly interesting morphologies and behaviors were evolved, the applicability of the system for evolving real robots remained questionable as the physics specifications for synthesizing the creatures allowed for interpenetrating surfaces. While Sims required Connection Machine CM-5 parallel computers to conduct his artificial evolution, Taylor and Massey (2001) recently re-implemented Sims' work using only standard personal computers. Ray (2000) has also implemented a system highly reminiscent of Sims' system but rather than using a fixed fitness function within the artificial evolutionary system for selection, he relied upon aesthetic user selection. Both Taylor and Massey (2001) and Ray (2000) used a commercial-off-the-shelf physics engine called MathEngine, which is the predecessor to the physics engine known as Vortex used in this thesis (see Section 3.1.1).

MathEngine was also used to develop a 3D biomechanical simulation of

a salamander for a computational neuroethological experiment into the underlying neural circuits that generated the aquatic and terrestrial locomotion of real salamanders (Ijspeert 2000). Although no evolutionary results were reported, an algorithmic CPG controller was developed that allowed for realistic and life-like swimming and trotting gaits to be reproduced in the artificial salamander. The CPGs were shown to be stable enough to receive higher-level sensory input from vision modules to enable tracking and approach towards a moving target (Ijspeert and Arbib 2000). These studies stem from earlier work on evolving CPGs based on neural controllers that generate swimming gaits for a 2D lamprey (Ijspeert, Hallam, and Willshaw 1999; Ijspeert and Kodjabachian 1999) as well as for generating locomotion gaits for a 2D salamander (Ijspeert, Hallam, and Willshaw 1998; Ijspeert 1999; Ijspeert 2001). The fitness of evolved neural controllers was evaluated using single-objective functions based on either single or multiply-combined network output metrics.

GP has also been used to evolve controllers for a fixed morphology virtual creature (Gritz and Hahn 1997). In this study, the emphasis was on evolving different control programs for a 3D animated character as opposed to traditional "key-framing" techniques that involved human hand-designed frames used by most animators. It was claimed that the evolved controller produced fluid, physically and biologically plausible motions. An incremental approach where additional constraints were phased into the single-objective fitness function as the evolutionary optimization progressed was used to evolve the final desired behavior. This incremental methodology was adopted after it was found that a direct approach incorporating all the desired motion styles into the fitness function from the start severely restricted the evolvability of the system. Both evolutionary robotics and virtual embodied evolution techniques have also been extended to practical applications in the entertainment and edutainment industries (Taylor 2000; Grand 2001; Lund 2001). Commercially-based research conducted by Grand (2001) produced the artificial life game called *Creatures* in which owners could breed, nurture and evolve virtual organisms known as *Norns* on their personal computers and even exchange genetic material with other owners over the Internet. Techniques used included

user-guided behavior-based systems, user-guided evolutionary and co-evolutionary robotics as well as those including morphogenesis (Lund 2001) while it was suggested that to enable scaling to more complex behaviors required for characters in computer games, other techniques such as lifetime learning, virtual ecologies and evolution of behavioral primitives need to be considered (Taylor 2000).

Bongard and Paul (2000) investigated the relationship between morphological symmetry and locomotive efficiency by co-evolving the controller and morphology of virtual embodied organisms using a physically accurate simulation. A variable length GA based on the SAGA algorithm (Harvey 1992) was used to evolve the artificial creatures and a recurrent neural architecture was used to act as the creatures' controllers. Two single-objective fitness functions were designed to reward firstly a combination of locomotion distance and morphological symmetry, and secondly locomotion distance and morphological asymmetry. It was found that bilaterally symmetrical agents were favored by evolution in terms of locomotion capability. Although these experiments entailed two separate objectives of distance and symmetry, these objectives were combined into a single fitness evaluation function.

Bongard and Pfeifer (2002) also conducted experiments in which only the weights for fixed architecture recurrent neural controllers were evolved. 10 creatures with different but fixed morphologies were used to investigate the difficulties of evolving locomotion controllers for creatures with different body masses and number of legs. Using a single-objective fitness function that measured forward displacement, it was claimed that hexapedal agents were the easiest while worm-like agents were the hardest to evolve successful controllers. In a related study, artificial evolution was shown to automatically add more complex behaviors to simpler ones through the use of different sensor modalities (Bongard 2002a). Using a quadrupedal agent with two simulated chemical sensors, a lower-level chemotaxis behavior was shown, through a number of lesioning experiments, to provide a base for the generation of a higher-level forward locomotion behavior. Hidden units in the neural networks were also lesioned to demonstrate that over evolutionary time, some hidden units became specialized in processing certain input signals.

A similar but separate series of studies focused on the developmental processes associated with evolving virtual embodied organisms where both the controller and morphology were again being co-evolved (Bongard and Pfeifer 2001; Bongard 2002b). Using the designed system called *Artificial Ontogeny* (AO), artificial organisms were evolved to locomote and push boxes in which a standard GA using a single-objective evaluation function was augmented with a genetic model based on biological differential gene expression. As such, the genotype-to-phenotype morphogenesis allowed for changing pattern expressions similar to that found in genetic regulatory networks (GRNs). It was claimed that the AO system had high evolvability since the artificial evolutionary system was able to produce modular structures as well as dissociate between the genotypic and phenotypic complexities (Bongard and Pfeifer 2001). In the later study, Bongard (2002b) showed that the AO system was able to generate modular GRNs early during the evolutionary process which led to the successful generation of creatures with high parts count. The early appearance of modular GRNs was attributed to the high pleiotropy (co-regulation of genes) within the neurogenesis process and low pleiotropy between the neurogenesis and morphogenesis processes. However, a somewhat biased weighted sum fitness function that involved a "shaping" term, which explicitly rewarded organisms with number of body parts, sensors, motors and synapses, was used to encourage the early appearance of active agents during evolution.

Komosinski and Ulatowski (1999) developed a proprietary platform for studying the evolution of 3D physically simulated virtual creatures called *Framsticks*. The system allows for both directed as well as open-ended evolutionary runs to be conducted although published results have only documented experiments with directed evolution for behaviors such as walking and swimming (Komosinski and Ulatowski 1999; Komosinski 2000). An initial investigation into the design and use of more evolvable genotype representations for achieving open-ended evolution was discussed by Komosinski and Rotaru-Varga (2000). It was found in a later study that higher-level encodings that included either recurrent or developmental elements in the genotype representation allowed for more structured phenotypes to be gen-

erated, which in turn led to the appearance of fitter individuals for separate single-objective maximization tasks involving height and locomotion speed (Komosinski and Rotaru-Varga 2001). The *Framsticks* artificial evolutionary system has also been used to create a method for studying the taxonomy of evolved agents based on how dissimilar agents were in terms of their morphological geometry (Komosinski and Kubiak 2001). The results obtained from using these taxonomic measures on artificial organisms were later compared to the characteristics and properties of biological phylogenic trees constructed for real organisms (Komosinski, Koczyk, and Kubiak 2001). The *Framsticks* creatures have also been used by Mandik (2002) to study the evolvability of mental representations where the neural controllers of fixed morphology agents were optimized using a combination of both human and artificial evolutionary design inputs for food-finding tasks in both walking and swimming creatures.

The emphasis of Hornby and Pollack (2001a) in their study of evolving both the controller and morphology of virtual creatures was also on the genotype encoding for achieving more complex designs. Using a developmental grammar based on Lindenmayer systems (L-systems), 3D agents with simple bars and actuators were evolved in a quasi-static virtual world which could physically simulate low momentum movements similar to that of Lipson and Pollack (2000). A weighted sum fitness function was utilized to optimize maximization of locomotion distance and minimization of occurrences where body parts were dragged on the ground. They showed that creatures evolved using generative encodings outperformed those evolved using non-generative encodings for a locomotion task by capturing useful design space biases while allowing large scale mutations to be performed viably, which in turn enabled the encapsulated and coordinated re-use of hierarchies of parts (Hornby and Pollack 2002). It was claimed that the morphologies of these generatively encoded creatures were more complex than those previously reported by Sims (1994b), Komosinski and Rotaru-Varga (2000) and Lipson and Pollack (2000), by virtue of having more parts in the morphology and more regularity in the overall design of the evolved creatures. In related work, oscillator controllers

similar to CPGs were used in place of neural network controllers for evolving both 2D (Hornby, Lipson, and Pollack 2001) and 3D virtual agents (Hornby and Pollack 2001b). The 2D agents were also successfully transferred to real physical robots (Hornby, Lipson, and Pollack 2001).

2.5 The Emergent Questions

As we have seen, the research into evolving artificial creatures have focused mainly on generating the desired behavior using single-objective fitness functions. These evaluation functions typically consist only of a single term for assigning the fitness of individuals generated (Sims 1994b; Arnold 1997; Gritz and Hahn 1997; Harvey, Husbands, Cliff, Thompson, and Jakobi 1997; Reeve 1999; Lipson and Pollack 2000; Fujii, Ishiguro, Aoki, and Eggenberger 2001; Komosinski and Rotaru-Varga 2001; Paul and Bongard 2001; Reil and Massey 2001; Bongard and Pfeifer 2002) or a combination of multiple terms into a single weighted objective when the desired behavior cannot be achieved with simpler single-termed functions (Lee, Hallam, and Lund 1996; Floreano and Mondada 1998; Husbands, Smith, Jakobi, and O'Shea 1998; Keymeulen, Iwata, Konaka, Suzuki, Kuniyoshi, and Higuchi 1998; Dittrich, Skusa, Banzhaf, and Kantschik 1999; Bongard and Paul 2000; Hornby and Pollack 2001a; Pasemann, Steinmetz, Hulse, and Lara 2001b; Bongard 2002b; Reil and Husbands 2002). It is highly surprising that a true multi-objective optimization approach involving optimization of explicitly distinct objectives have not been explored yet thus far for artificial creature evolution. Such an investigation might very well reveal significant advantages over standard single-objective EAs in terms of the evolutionary optimization process itself in addition to the possibility of generating greater varieties of creature morphologies and behaviors. We investigate this problem in Chapters 5, 6 and 8.

Although it was reported that more complex creatures could be evolved with certain artificial evolutionary systems (Hornby and Pollack 2001a; Komosinski and Rotaru-Varga 2001; Bongard 2002b), these claims were made simply based on

the fact that the artificial creatures had more moving parts or greater regularity in their morphology. Obviously such a trivial comparison leaves much to be desired since a millipede would be considered to be more complex than a human on both counts! Hence, it remains unclear how we can objectively compare between the complexities of artificially evolved creatures using more intuitive measures or methodologies. We attempt to tackle this problem in Chapter 7.

Additionally, apart from the work of Smith with wheeled robots (2001b, 2001a, 2002), there has been little effort invested in systematic explorations of the fitness landscape characteristics for evolving artificial creature controllers. There has been even less work conducted on characterizing the underlying search space difficulty when evolving controllers for legged artificial creatures. Only the work of Dittrich, Skusa, Banzhaf, and Kantschik (1999) with an abstract morphology robot has provided some empirical information regarding the evolutionary fitness landscape for a non-wheeled robot. Although a conjecture concerning the smoothness of the underlying search space for evolving CPG controllers for bipedal robots was postulated by Reil and Husbands (2002), no actual experimental results have been reported yet thus far with regards to testing this hypothesis. As such, very little is known at this stage concerning the fitness landscape and difficulty associated with evolutionary searching of controllers for legged artificial creatures. We attempt to tackle this problem in Chapter 4.

2.6 Chapter Summary

A literature review of the related fields of evolutionary robotics and evolution of embodied artificial life was presented in this chapter. The importance of situating and embodying the artificial agents within a physically-based world was first highlighted. A comprehensive survey of the various methods employed for evolving artificial creatures comprising of both real physical robots and simulated virtual agents was then given. Finally, the research questions emerging from this literature review were presented.

Chapter 3

The Virtual World

This chapter outlines the virtual world used for conducting the experiments to be presented in this thesis. It is divided into five main sections. First, we briefly discuss the physics engine used to simulate the creature and its environment. Next, we explain in detail the setup of the creature's morphology. Then we describe the basic workings of an artificial neural network (ANN) that act as the creature's locomotion controller, as well as evolutionary methods of learning for ANNs. The next section then explains the genotype used to represent the artificial creature's locomotion controller including an outline of the process which converts the genotype into the operational ANN used to control the movement of the artificial creature. Finally we discuss the choice of our common evolutionary, simulation parameters and statistical test of significance used for the experiments conducted in this thesis.

3.1 Physics-Based Simulation

The accurate modelling of the simulation environment plays a crucial part in producing artificial creatures that move and behave realistically in 3D (Taylor and Massey 2001). A dynamic rather than kinematic approach is paramount in allowing for effective artificial evolution to occur. Physical properties such as forces, torques, inertia, friction, restitution and damping need to be incorporated into the artificial evolutionary system. To this end, the Vortex physics engine (CM Labs 2002) was

employed to generate the physically realistic artificial creature and its simulation environment.

By virtue of conducting our artificial evolution within a physically accurate virtual world, rich dynamical interactions are able to occur between the simulated creature and its environment. This in turn enables complex walking behaviors to emerge as the creature evolves the use of its sensors to control the actuators in its limbs through dynamical interactions with the environment. Furthermore, the accurate modelling of physical forces results in creature locomotion that is both realistic and biologically interesting.

3.1.1 Vortex Physics Engine

The Vortex physics engine is a commercial-off-the-shelf simulation toolkit developed by Critical Mass Laboratories (CM Labs 2002). It consists of a set of libraries that comprise of C^{++} routines for robust rigid-body dynamics, collision detection, contact creation, and collision response. Vortex is currently being used by NASA's Autonomy and Robotics Group to develop autonomous rovers for possible deployment in future Mars exploration programs (CM Labs Press Release 2003). The advantages of Vortex include allowing the simulation of natural behavior of objects in the physical world and offering fast yet accurate computation of the dynamical forces that act on the simulated objects. Using the supplied libraries, real-time interactive 3D simulations can be created in which objects, particularly jointed objects moving under constraints, exhibit natural movement under various environmental conditions. However, as Vortex is a constraint-based simulation, it naturally suffers from increasingly higher computational requirements as the number of objects being simulated in the world increases. As such, the design of the artificial creature and its world are kept relatively simple in order to maintain a reasonable run time, especially when conducting the evolutionary experiments.

3.2 Creature Morphology

The artificial creature is a basic quadruped with 4 short legs. Each leg consists of an upper limb connected to a lower limb via a hinge (one degree of freedom) joint and is in turn connected to the torso via another hinge joint. Therefore, there are 8 degrees of freedom overall. The mass of the torso is 1g and each of the limbs is 0.5g. The torso has dimensions of $4 \times 2 \times 1$cm and each of the limbs has dimensions of $1 \times 1 \times 1$cm. In terms of its morphological dimensions, the creature can be visualized as some type of insect. A biological equivalent in terms of size and weight can be found in beetles, where their body lengths range from 0.25mm to 16cm and body mass from 0.4mg to 30g (Grzimek 1984). Research into evolving tiny creatures is being given more attention lately, especially in the field of nanotechnology. A screen dump of the actual creature within the simulation world is shown in Figure 3.1 and a geometric representation of the creature is given in Figure 3.2.

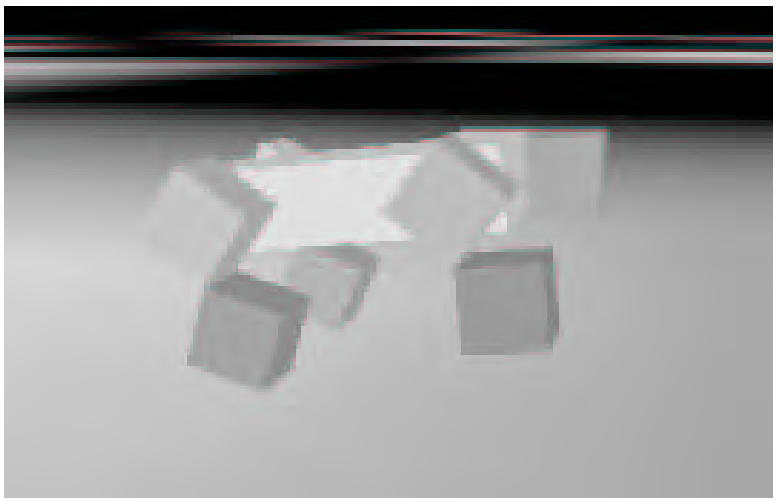


Figure 3.1: A screen dump of the simulated quadruped in the simulation world.

The hinge joints are allowed to rotate between 0 to 1.57 radians. Each of the hinge joints is actuated by a motor that generates a torque producing rotation of the connected body parts about that hinge joint. The creature's overall central nervous system is illustrated in Figure 3.3 where the three letter abbreviations identify each of the 8 different limbs. The first letter denotes (**U**)pper or (**L**)ower, the second denotes to (**F**)ront or (**B**)ack, and the third denotes (**R**)ight or (**L**)eft.

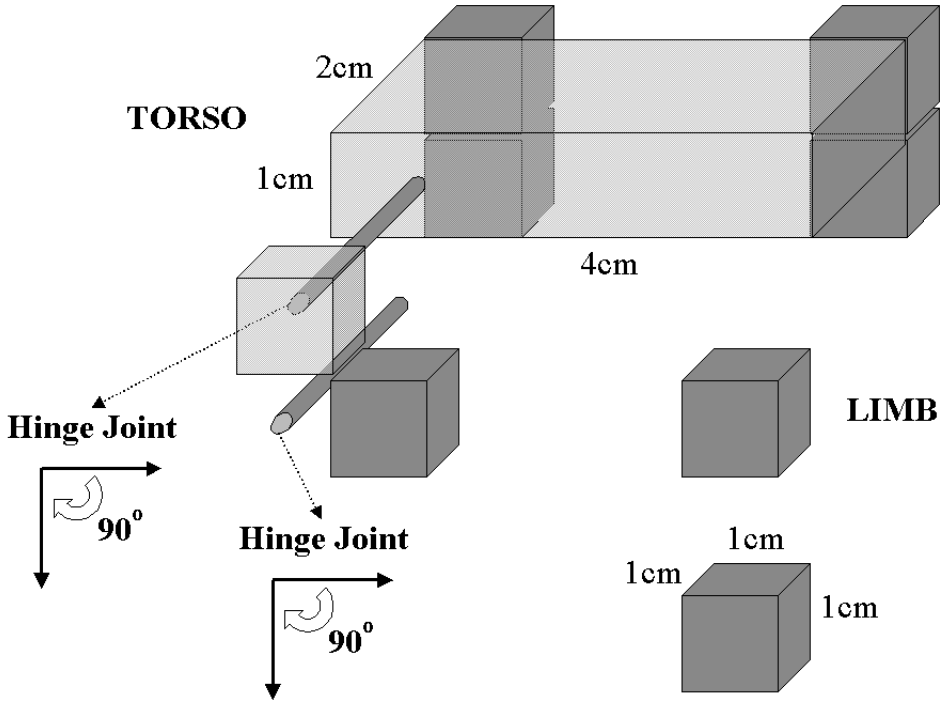


Figure 3.2: A geometric representation of the simulated quadruped.

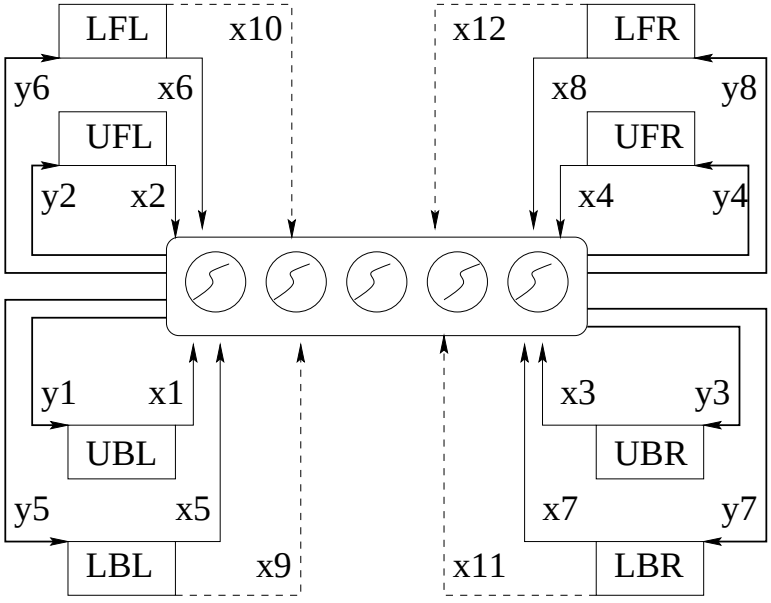


Figure 3.3: A diagrammatic representation of the simulated quadruped's central nervous system.

Correspondingly, the artificial creature has 12 sensors and 8 actuators. The 12 sensors consist of 8 joint angle sensors $(x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8)$ corresponding

to each of the hinge joints and 4 touch sensors ($x_9, x_{10}, x_{11}, x_{12}$) corresponding to each of the 4 lower limbs of each leg. The joint angle sensors (x_1–x_8) return continuous values in radians whereas the touch sensors (x_9–x_{12}) return discrete values, 0 if no contact with the ground and 1 if contact is made. The choice of inputs was decided upon after a review of the literature where joint angle and touch sensors were the most commonly used types of input to the controller (eg. Sims 1994a; Komosinski 2000; Paul and Bongard 2001; Bongard and Pfeifer 2002). The 8 actuators ($y_1, y_2, y_3, y_4, y_5, y_6, y_7, y_8$) represent the motors that control each of the 8 articulated joints of the creature. These motors are controlled via outputs generated from the ANN controller which is then used to set the desired velocity of rotation of the connected body parts about that joint. A summary of the sensors is given in Table 3.1 followed by the actuators in Table 3.2.

Sensor Number	**Detects Angle Between**	**Value Returned**
x_1	Torso & upper back left limb	$[0, 1.57]$
x_2	Torso & upper front left limb	$[0, 1.57]$
x_3	Torso & upper back right limb	$[0, 1.57]$
x_4	Torso & upper front right limb	$[0, 1.57]$
x_5	Upper & lower back left limbs	$[0, 1.57]$
x_6	Upper & lower front left limbs	$[0, 1.57]$
x_7	Upper & lower back right limbs	$[0, 1.57]$
x_8	Upper & lower front right limbs	$[0, 1.57]$
Sensor Number	**Detects Contact Between**	**Value Returned**
x_9	Lower back left limb & ground	$\{0, 1\}$
x_{10}	Lower front left limb & ground	$\{0, 1\}$
x_{11}	Lower back right limb & ground	$\{0, 1\}$
x_{12}	Lower front right limb & ground	$\{0, 1\}$

Table 3.1: Description of the simulated quadruped's 12 sensors that provide inputs to the ANN controller.

3.3 Creature Controller

3.3.1 Artificial Neural Networks

An ANN may be described as a directed graph: $G(N, A, \psi)$, where N is a set of nodes, A denotes the connections between the nodes, and ψ represents the

Actuator Number	Controls Joint Between	Velocity Range
y_1	Torso & upper back left limb	$[-5, +5]$
y_2	Torso & upper front left limb	$[-5, +5]$
y_3	Torso & upper back right limb	$[-5, +5]$
y_4	Torso & upper front right limb	$[-5, +5]$
y_5	Upper & lower back left limbs	$[-5, +5]$
y_6	Upper & lower front left limbs	$[-5, +5]$
y_7	Upper & lower back right limbs	$[-5, +5]$
y_8	Upper & lower front right limbs	$[-5, +5]$

Table 3.2: Description of the simulated quadruped's 8 actuators that receive outputs from the ANN controller.

learning rule which enables the strengths of inter-neuron connections to be automatically adjusted. A node receives its inputs from an external source or from other nodes in the network. The node undertakes some processing on its inputs and sends the result as its output. The processing function of a node is called the activation function. The activation, a, is calculated as a weighted sum of the inputs to the node in addition to a constant value called the bias.

The following notations will be used to describe a single hidden layer feed-forward ANN:

- I and H are the number of input and hidden units respectively.
- $\mathbf{X}^p \in \mathbf{X} = (x_1^p, x_2^p, \ldots, x_I^p), p = 1, \ldots, P$, is the pth pattern in the input feature space $\mathbf{X}$ of dimension I, and P is the total number of patterns.
- $\mathbf{Y}_o^p \in \mathbf{Y_o}$ is the corresponding scalar of pattern $\mathbf{X}^p$ in the output target space $\mathbf{Y_o}$.
- w_{ih} and w_{ho}, are the weights connecting input unit i, $i = 1, \ldots, I$, to hidden unit h, $h = 1 \ldots H$, and hidden unit h to the output unit o.
- $\Theta_h(\mathbf{X}^p) = \sigma(a_h)$; $a_h = \sum_{i=0}^{I} w_{ih} x_i^p$, $h = 1, \ldots, H$, is the h^{th} hidden unit's output corresponding to the input pattern $\mathbf{X}^p$, where a_h is the activation of hidden unit h, and $\sigma(.)$ is the activation function that is taken in this paper to be the logistic function $\sigma(z) = \frac{1}{1+e^{-Dz}}$, with D the function's sharpness or steepness and is taken to be 1.

- $\hat{Y}_o^p = \sigma(a_o)$; $a_o = \sum_{h=0}^{H} w_{ho}\Theta_h(\mathbf{X}^p)$ is the network output and a_o is the activation of output unit o corresponding to the input pattern $\mathbf{X}^p$.

The following pseudocode describes the functioning of a single hidden layer feed-forward ANN in operation.

1. Until termination conditions are satisfied, do
   (a) for each input pattern, $(\mathbf{X}^p, Y_o^p)$, apply the following steps
      i. Inject the input pattern $\mathbf{X}^p$ into the network.
      ii. Calculate the output, $\Theta_h(\mathbf{X}^p)$, for each hidden unit h.
      iii. Calculate the output, $\hat{Y_o}^p$, for each output unit o.

3.3.2 Evolutionary Artificial Neural Networks

Traditionally ANNs are trained using learning algorithms such as *back-propagation* (BP) (Rumelhart, Hinton, and Williams 1986) to determine the connection weights between nodes. However such methods are gradient-based techniques which usually suffer from the inability to escape from local minima when attempting to optimize the connection weights. To overcome this problem, evolutionary approaches have been proposed as an alternative method for optimizing the connection weights. ANNs evolved via this method is thus referred to as evolutionary ANNs (EANNs). In the literature, research into EANNs usually involves one of three approaches: (1) evolving the weights of the network (Fogel, Fogel, and Porto 1990; Belew, McInerney, and Schraudolph 1992), (2) evolving the architecture (Miller, Todd, and Hegde 1989; Kitano 1990), or (3) evolving both simultaneously (Koza and Rice 1991; Angeline, Saunders, and Pollack 1994). For a thorough review of EANNs, refer to the comprehensive survey conducted by Yao (1999).

Our objective is to evolve ANNs that can perform successfully as locomotion controllers for artificial creatures. Here we will attempt to optimize both the connection weights and number of hidden nodes through evolutionary methods. Other architectural aspects of the ANN such as types of connections between layers,

types of transfer functions and number of input/output units have been kept fixed and are not subjected to evolutionary optimization.

3.3.3 Controller Architectures

The choice of ANN architectures used for controller evolution is normally made arbitrarily when evolving both simulated (Reeve 1999; Bongard and Paul 2001; Pasemann, Steinmetz, Hulse, and Lara 2001a; Paul and Bongard 2001; Reil and Massey 2001; Bongard 2002a; Bongard and Pfeifer 2002; Reil and Husbands 2002) as well as physical artificial creatures (Husbands, Harvey, Jakobi, Thompson, and Cliff 1997; Lund and Hallam 1997; Floreano and Urzelai 1998; Floreano and Mondada 1998; Nolfi and Floreano 1999; Floreano and Urzelai 2000; Nolfi and Floreano 2002; Nolfi 2002). Usually some form of recurrency is used in the ANN, either partially (Bongard and Paul 2001; Pasemann, Steinmetz, Hulse, and Lara 2001a; Paul and Bongard 2001; Bongard and Pfeifer 2002; Bongard 2002a) or fully (Floreano and Urzelai 1998; Floreano and Mondada 1998; Nolfi and Floreano 1999; Reeve 1999; Floreano and Urzelai 2000; Reil and Massey 2001; Nolfi and Floreano 2002; Nolfi 2002; Reil and Husbands 2002). On the other hand, simple direct connections between sensor inputs and motor outputs have also proven to be sufficient for evolving robots controllers with simple behaviors that can accomplish the set task (Lund and Hallam 1997; Pasemann, Steinmetz, Hulse, and Lara 2001a; Nolfi 2002). As such, it remains unclear from the body of literature what types of ANN architecture should be used to evolve controllers for artificial creatures.

We will now introduce four different types of ANN architecture for controller evolution where in the next chapter, we will attempt to provide some characterization of the search space associated with each type of controller architecture. The first type of ANN is a simple feed-forward ANN and is denoted NNType0 (Fig. 3.4.1). The second type of ANN is a feed-forward ANN augmented with direct connections from input to output units and is denoted NNType1 (Fig. 3.4.2). The third type of ANN is a feed-forward network with recurrency on the hidden units (Elman-type ANN (Elman 1990)) and is denoted NNType2 (Fig. 3.4.3). The last

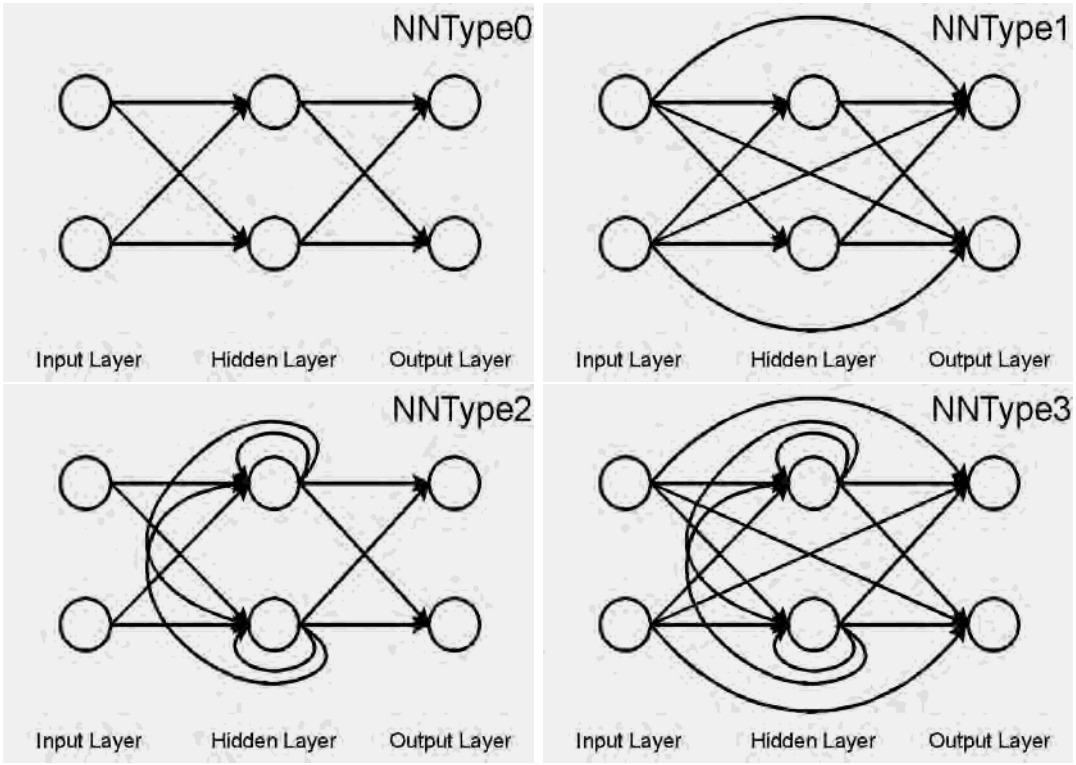


Figure 3.4: Diagrammatic representation of 4 types of ANN architecture: 1. NNType0 (top left), 2. NNType1 (top right), 3. NNType2 (bottom left), 4. NNType3 (bottom right).

type of ANN is a feed-forward network augmented with both direct connections from input to output units as well as recurrency on the hidden units and is denoted NNType3 (Fig. 3.4.4).

Recurrent connections were included to allow the creature's controller to learn state-dependent dynamics of the system. Direct input-output connections were also included in the controller's architecture to allow for direct sensor-motor mappings to evolve that do not require hidden layer transformations. For all four ANN architectures, all units in the preceding layer are fully-connected to all units in the following layer. A bias term is also incorporated into the calculations of the activations of the hidden as well as output units.

All four ANN architectures have a variable hidden layer in terms of its number of active hidden units. This is an integral feature of the ANNs that is essen-

tial for the purposes of this investigation as explained in Section 1.2. Experiments and analyses found later in this thesis will further clarify this point.

3.4 Genotype Representation

Our chromosome is a class that contains one matrix Ω and one vector ρ. The matrix Ω is of dimension $(I + H) \times (H + O)$. Each element $\omega_{ij} \in \Omega$, is the weight connecting unit i with unit j, where $i = 0, \ldots, (I - 1)$ is the input unit i, $i = I, \ldots, (I + H - 1)$ is the hidden unit $(i - I)$, $j = 0, \ldots, (H - 1)$ is the hidden unit j, and $j = H, \ldots, (H + O - 1)$ is the output unit $(j - H)$.

The vector ρ is of dimension H, where $\rho_h \in \rho$ is a binary value used to indicate if hidden unit h exists in the network or not. As such, it works as a switch to turn a hidden unit on or off. The sum, $\sum_{h=0}^{H} \rho_h$, represents the actual number of hidden units in a network, where H is the maximum number of hidden units. The use of ρ allows a hidden node to evolve even if it is not active during certain periods of the evolutionary optimization process.

The chromosome has two additional components when the crossover and mutation rates are also subjected to evolutionary optimization and self-adapted in the algorithms. These additional elements are the crossover rate δ and the mutation rate η. The addition of these last two elements to the genotype representation allows simultaneous training of the weights in the network and selecting a subset of hidden units as well as allowing for the self-adaptation of crossover and mutation rates during optimization.

A direct encoding method was chosen to represent these variables in the genotype as an easy-to-implement and simple-to-understand encoding scheme. Other more complex direct as well as indirect encoding schemes such as those involving developmental mechanisms may prove to be useful and represents possible future work extending from this investigation. A summary of the variables used in the chromosome to represent the artificial creature's genotype is listed in Table 3.3. The mapping of the chromosome into the ANN is depicted in Figure 3.5.

Variable	Representing	Value Type	Value Range
Ω	ANN Connection Weights	Real	$]-\infty,+\infty[$
ρ	Active Hidden Units	Discrete	$\{0,1\}$
δ	Crossover Rate *	Real	$[0,1]$
η	Mutation Rate *	Real	$[0,1]$

Table 3.3: Description of the variables used in the chromosome to represent the artificial creature's genotype. * denotes elements present only in algorithms that use self-adaptation of crossover and mutation rates.

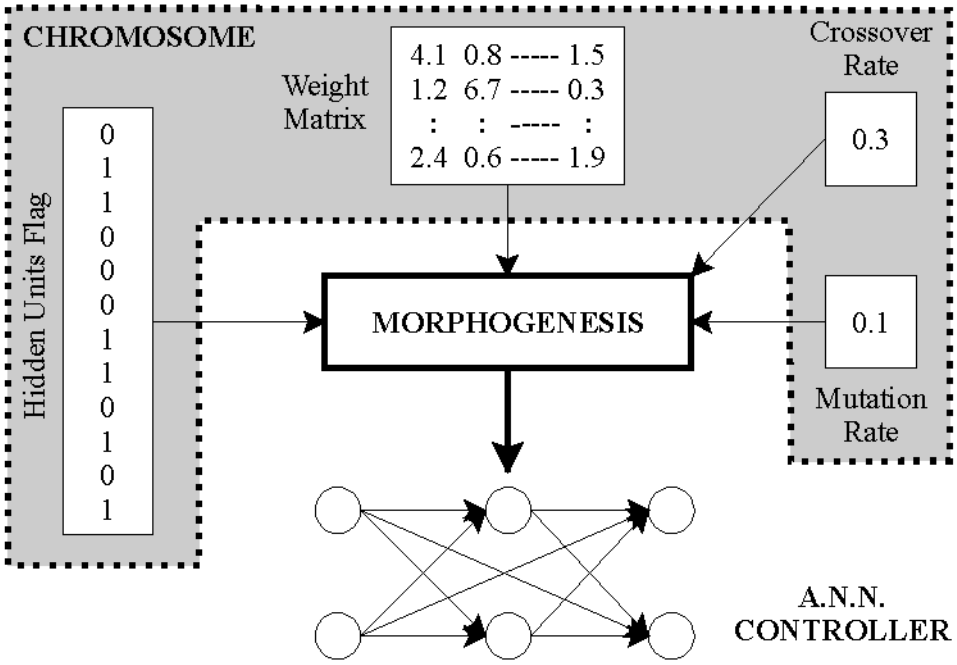


Figure 3.5: A diagram illustrating the mapping from a chromosome to an ANN controller.

3.4.1 Fitness Functions

The fitness f_1 of each locomotion controller represented in the genotype g is defined to be simply

$$f_1 = \Uparrow \ d(g) \tag{3.1}$$

where d refers to the horizontal Euclidean distance achieved by the creature as controlled by the ANN at the end of the evaluation period of 500 timesteps. In other

words, d measures the planar locomotion distance travelled by the artificial creature over the designated period. In experiments involving multi-objective evolutionary optimization of the locomotion controller, a second fitness f_2 is defined to be

$$f_2 = \Downarrow \sum_{h=0}^{H} \rho_h \tag{3.2}$$

where the number of active hidden units in used in the ANN controller is counted by summing the binary vector ρ_h, which is part of the genotype g. Therefore, the first objective is to maximize the horizontal locomotion achieved by the artificial creature and where a second objective is involved, to minimize the use of nodes in the hidden layer of the ANN controller, which will in turn determine the VC-dimension of the controller as explained in Section 1.3. Unless stated explicitly, the fitness of a controller always refers to the first fitness function f_1 which measures the locomotion distance.

3.5 Evolutionary and Simulation Parameters

The relevant literature on artificial evolution of physically simulated creatures was reviewed to ascertain suitable parameter values for use in the evolutionary runs to be conducted in the experiments. Table 3.4 summarizes the parameters used for these types of experiments.

Authors	No. of Generations	Population Size	No. of Timesteps	No. of Repeats
Lipson and Pollack (2000)	300–600	200	12	N/A
Bongard and Paul (2000)	300	300	20000	10
Hornby and Pollack (2001a)	100–500	100	N/A	10
Taylor and Massey (2001)	50–100	∼300	200–1000	N/A
Otsu, Ishiguro, Fujii, Aoki, and Eggenberger (2001)	500	100	N/A	5
Komosinski and Rotaru-Varga (2001)	120–800	200	50–500	10
Bongard (2002a)	50	200	500	10
This Thesis	1000	30	500	10

Table 3.4: A comparison of the evolutionary and simulation parameters used for evolving artificial creatures in simulation.

As can be seen, the parameters used in the literature varied significantly. The evolutionary parameters used in this thesis were chosen after both reviewing the literature and conducting a number of preliminary experiments. This was to ensure that the evolutionary runs balanced both time requirements and quality of solutions obtained. These values are given in the last line of Table 3.4. It may be noticed that the number of generations we have chosen is above the normal range used in the literature. This is to ensure that the evolutionary optimization has been given enough time to converge to a reasonably optimal solution. Secondly, the use of a small population size is also noted with our proposed multi-objective evolutionary optimization algorithm. Small population sizes have been previously shown to be advantageous when evolving ANNs using GAs (Foster, McCullagh, and Whitford 1999). Furthermore, preliminary experiments have shown that a large population size was not essential for evolving successful locomotion controllers (Teo and Abbass 2002a; Teo and Abbass 2002b; Teo and Abbass 2002c). One reason is that the objective of minimizing the number of hidden units which is discrete in nature imposes an upper bound on the possible number of non-dominated solutions that can exist in the population (see Section 5.1 for an explanation of non-dominance and Pareto optimality). This upper bound is simply the maximum number of hidden units allowed + 1 (see second paragraph of Section 4.4 for further explanations). A population size of 30 is approximately double the bound on the number of non-dominated solutions that can exist in any population using this approach (Abbass 2002a). Since our approach is an elitist Pareto approach, we only preserve the non-dominated set in each population. Therefore, we do not need to introduce additional methods for reducing the number of Pareto solutions when they exceed a certain threshold as in the case of the PDE algorithm (Abbass, Sarker, and Newton 2001) where a neighborhood function was used, and NSGA-II (Deb, Agrawal, Pratab, and Meyarivan 2000) where a niching strategy was used.

3.5.1 Statistical Testing

In order to test for the presence of significant statistical differences between two sets of results, the paired two-sample t-test (Runyon, Haber, Pittenger, and Coleman 1996) is used throughout all experimental setups in this thesis. Since the main objective of this work is to investigate the automatic generation of locomotion controllers, all tests of significance are always conducted in relation to the f_1 objective which evaluates locomotion fitness. The t-statistic is calculated using the direct difference method (Runyon, Haber, Pittenger, and Coleman 1996) which is given by the equation

$$t = \frac{\bar{D}}{\sqrt{\frac{\sum D^2 - \frac{(\sum D)^2}{N}}{N(N-1)}}} \tag{3.3}$$

where $\bar{D} = \bar{X}_1 - \bar{X}_2$ and $\bar{X}_1$, $\bar{X}_2$ are the sample means of the two groups and D is the difference between the corresponding pairs of random variables. The 10 different initial populations are fixed for all experiments by using the same set of 10 seeds corresponding to each of the 10 individual runs. N is the sample size of a single group and the degree of freedom is approximated by taking $N - 1$. For all statistical testing of results, a two-tailed test at both significance levels of $\alpha = 0.05$ (95% confidence interval, t-value = 2.262) and $\alpha = 0.01$ (99% confidence interval, t-value = 3.250) are conducted. Bracketed t-values indicate that the sample mean of the group being tested, X_2, is lower compared to the sample mean group of results tested against, X_1.

In experiments involving comparisons between different ANN architecture types, the tests for statistical significance are always made against the NNType0 architecture. As explained earlier in Section 3.3.3, the NNType0 architecture is a simple feed-forward artificial neural network with connections only between the input-hidden and hidden-output layers. It does not have any other additional direct nor recurrent connections that are present in the other types of controller architectures. Thus, the NNType0 controllers have the most minimal architectures among

the different types of controller architectures assuming that the number of hidden units is fixed. Hence, to test whether more complex types of ANN architectures with additional connections provided significantly different locomotion capabilities, t-tests are always carried out against the NNType0 architecture.

3.6 Chapter Summary

The design of our physically-based artificial evolutionary system for evolving the locomotion controllers of a virtually embodied creature was described. The choice of parameters to be used in the experimental sections of this thesis were also discussed and justified. In the next chapter, we will attempt to characterize the fitness landscapes of four ANN architectures to investigate which type of network will be most suitable for the artificial evolution of controllers.

Chapter 4

Fitness Landscapes

[1] As we have seen from Chapter 2, there has been a lot of interest in evolving controllers for both physically simulated creatures as well as for real physical robots. However, a range of different ANN architectures are used for controller evolution and in the majority of the work conducted, the choice of the architecture used is made arbitrarily.

There have been some preliminary experiments that compared the performance of evolved feed-forward versus dynamically recurrent ANNs for controlling Khepera robots in a simulated space-constrained box-pushing experiment (Spronck, Sprinkhuizen-Kuyper, and Postma 2001). It was found that feed-forward architectures were sufficient for successfully completing the task although the recurrent architecture provided much more stability for the controller's behavior, particularly in maneuvering out of problematic positions. However, no fitness landscape analysis was provided for the underlying fitness landscape of the controller's search space.

As such, the literature remains largely inconclusive as to which ANN architecture provides the most efficient and effective space for searching the range of possible controllers through evolutionary methods. This represents the motivation for this chapter where we compare the search space for four different types of ANN architecture for controller evolution through an analysis of the fitness landscape as-

[1] Some of the material presented in this chapter have been previously published in Teo and Abbass (2003).

sociated with each type of architecture. We intend to ascertain whether additional recurrent and input-output connections to a standard feed-forward ANN architecture yields any benefit in terms of providing a fitness landscape which is easier to search for locomotion controllers. The results from this chapter will provide a basis for selecting the appropriate ANN architecture to be used for controller evolution in later chapters of this thesis.

4.1 Introduction

The idea of fitness landscapes was first proposed by Wright (1932) and has since proven to be an invaluable tool for analyzing evolutionary theories (Kauffman 1993; Adami 1998). It serves as a powerful tool for visualizing the evolutionary process through its imagery of mountainous peaks, hills, valleys, ridges and plateaus that are encountered through the exploration and exploitation of genotype space. Evolution can thus be viewed as movements within a multi-dimensional search space. Although initially introduced by Wright as a non-mathematical tool for visualizing biological selection and variation, fitness landscapes have since become highly amenable to mathematical analysis. A discussion of the various metrics that have been proposed for mathematically characterizing fitness landscapes is given in Section 4.3.

In an evolutionary computation context, a fitness landscape comprises of three main elements: (1) the set of genotypes, (2) the fitness function that evaluates the genotypes, and (3) the genetic operators that define the neighborhood relationships between the set of genotypes (Vassilev, Fogarty, and Miller 2000). The fitness landscape is thus normally a high-dimensional space with $n + m$ dimensions, where n is the genotype length and the extra m dimensions representing the fitness values associated with the genotype when evaluated using the m fitness functions. In traditional evolutionary computation, m is usually 1 since the evolutionary optimization process is conducted on one objective function only. In this case, a genotype with length two can be visualized as a 3D landscape where genetic operations carried out

on the set of genotypes will produce small movements on the landscape during the evolutionary search process. On the other hand, in an evolutionary multi-objective search process, m would be ≥ 2 since the solution is being optimized along two or more objective functions.

4.2 Search Space Difficulty

The performance of an EA is thus tied intimately to the structure of its fitness landscape. In attempting to identify the difficulties presented by a particular landscape, the evolutionary search is typically characterized in terms of the degree of epistasis and modality (Smith, Husbands, Layzell, and O'Shea 2002). Epistasis refers to the situation where the fitness of a genotype is dependent on multiple gene interactions. Modality refers to the situation where the search space has large numbers of optima. Both high epistasis and modality will lead to a rugged fitness landscape (Vassilev, Fogarty, and Miller 2000). Such rugged search spaces can be visualized as a landscape with many hill-tops that are separated by deep valleys. In other words, there is no steady or smooth progression of fitness values from one point to another neighboring point and thus increases the difficulty for the evolutionary search to move to higher areas of fitness during the optimization process. Therefore, highly epistatic and multi-modal problems will lead to a rugged landscape that is more difficult to search compared to a smoother landscape with low epistasis and modality.

4.3 Analyzing Fitness Landscapes

A number of fitness landscape analysis techniques have been proposed for measuring the degree of ruggedness of the underlying search space. These mathematical treatments of the fitness landscape comprise of two main streams: (1) statistical measures, and (2) information measures.

4.3.1 Statistical Measures

Weinberger (1990) used the autocorrelation function to measure the ruggedness of the landscape. A sequence of fitness values is generated using a random walk through the search space. The autocorrelation ρ between sets of fitness points separated by a distance of Γ is then approximated by

$$\rho(\Gamma) \approx \frac{E(f_t f_{t+s}) - E(f_t)E(f_{t+s})}{V(f_t)} \tag{4.1}$$

where $E(f_t)$ represents the expectation and $V(f_t)$ the variance of the sequence of N fitness values $\{f_t\}_{t=1}^{N}$. A high correlation indicates a smooth landscape since neighboring points have highly similar fitness values. On the other hand, a low correlation indicates a rugged landscape since neighboring points have highly dissimilar fitness values.

Weinberger also proposed another correlation measure called the correlation length to define landscape ruggedness. It is simply the distance beyond which the sets of fitness points become uncorrelated. The correlation length τ between sets of fitness points separated by a distance of Γ is calculated as

$$\tau(\Gamma) = -\frac{1}{\ln(\rho(1))} \tag{4.2}$$

where $\rho(1)$ is the autocorrelation of neighboring points. The magnitude of this length indicates the smoothness of the landscape. A longer correlation length would thus indicate a very smooth fitness landscape whereas a shorter length would indicate a more rugged landscape. A number of other correlational metrics have also been proposed for characterizing fitness landscapes (Manderick, de Weger, and Spiessens 1991; Lipsitch 1991; Hordijk 1996; Vassilev, Fogarty, and Miller 2000).

4.3.2 Information Measures

Apart from statistical analysis, a distinctly different methodology based on classical information theory (Shannon 1948) known as information content has been proposed for characterizing fitness landscapes (Vassilev, Fogarty, and Miller 2000). It also approximates the ruggedness of the underlying search space through

analysis of a sequence of fitness values $\{f_t\}_{t=1}^{N}$ obtained through a random walk of N steps over the landscape but instead measures the entropy or amount of fitness change encountered during the walk. Four measures were proposed by Vassilev in conjunction with this information analysis of fitness landscapes (Vassilev, Fogarty, and Miller 2000):

1. *Information Content* ($H(\epsilon)$): indicates the ruggedness of landscape path
2. *Partial Information Content* ($M(\epsilon)$): indicates the modality of landscape path
3. *Information Stability* ($\epsilon^{\star}$): indicates the magnitude of landscape path's optima
4. *Density-Basin Information* ($h(\epsilon)$): characterizes the landscape structure around optima

The information content characterizes the amount of ruggedness with respect to the flat areas of the landscape. The degree of flatness depends on a sensitivity parameter ϵ which is explained in the following paragraph. The information content is given by

$$H(\epsilon) = -\sum_{p \neq q} P_{[pq]} \log_6 P_{[pq]} \tag{4.3}$$

where $H(\epsilon)$ represents the entropy of the system. The probabilities $P_{[pq]}$ represent the frequencies of possible sub-blocks pq of elements from the string $S(\epsilon) = s_1 s_2 s_3 ... s_N$ where $s_i \in \{\bar{1}, 0, 1\}$. The string $S(\epsilon)$ is enumerated using the following function

$$s_i = \Psi_{f_t}(i, \epsilon) \tag{4.4}$$

where

$$\Psi_{f_t}(i, \epsilon) = \begin{cases} \bar{1} & \text{if } f_i - f_{i-1} < -\epsilon \\ 0 & \text{if } |f_i - f_{i-1}| \leq \epsilon \\ 1 & \text{if } f_i - f_{i-1} > \epsilon \end{cases} \tag{4.5}$$

for a particular value of the parameter ϵ. This parameter ϵ controls the sensitivity for measuring the entropy and is a real-valued number chosen from the range $[0, L]$ where L represents the maximum fitness difference of the sequence $\{f_t\}_{t=1}^{N}$.

The information analysis will be most sensitive when ϵ is 0 producing the maximal string of 1's and $\bar{1}$'s when enumerating $S(\epsilon)$ and hence provides the most detailed description of the landscape. Conversely, the analysis will be least sensitive when ϵ is L producing a string of 0's when enumerating $S(\epsilon)$ and hence provides the least detailed description of the landscape. In effect, ϵ acts as an accuracy setting for the information analysis and provides an idea of the landscape profile according to varying degrees of detail.

The partial information content is obtained by filtering out non-essential parts of $S(\epsilon)$ to obtain an indication of the modality encountered during the walk. It is given by

$$M(\epsilon) = \frac{\mu}{n} \tag{4.6}$$

where μ is the length of the derived string $S'(\epsilon)$ and n is the length of the original string $S(\epsilon)$. μ is calculated using a recursive function $\Phi_S(1, 0, 0)$ defined as

$$\Phi_S(i, j, k) = \begin{cases} k & \text{if } i > n \\ \Phi_S(i+1, i, k+1) & \text{if } j = 0 \text{ and } s_i \neq 0 \\ \Phi_S(i+1, i, k+1) & \text{if } j > 0, s_i \neq 0 \text{ and } s_i \neq s_j \\ \Phi_S(i+1, j, k) & \text{otherwise} \end{cases} \tag{4.7}$$

where k will return the value of μ upon completion of the evaluation. When $M(\epsilon)$ is 0, this is an indication that no slopes were present in the path. However when $M(\epsilon)$ is 1, this indicates that the path is maximally multi-modal. Furthermore, the expected number of optima can be calculated from the partial information content as $\lfloor \frac{nM(\epsilon)}{2} \rfloor$.

The information stability ($\epsilon^\star$) is defined to be the smallest value of ϵ corresponding to $H(\epsilon) = 0$. A high information stability indicates that the largest possible difference between two neighboring points is similarly high. Thus, it provides an idea of the magnitude of the landscape path's optima encountered during the walk.

In order to characterize the landscape structure around optima, it was also suggested that the *density-basin information* ($h(\epsilon)$) be calculated (Vassilev, Fogarty,

and Miller 2000). This measure gives an indication of the flat and smooth areas of the landscape and gives an indication of the density as well isolation of peaks in the landscape. The formula for calculating this measure is given by

$$h(\epsilon) = -\sum_{p \in \{\bar{1},0,1\}} P_{[pp]} \log_3 P_{[pp]} \tag{4.8}$$

where pp represent sub-blocks 00,11 and $\bar{1}\bar{1}$. A high number of peaks existing within a small area of the landscape would thus give a high value for $(h(\epsilon))$. On the other hand, an isolated optimum would give a low value for $(h(\epsilon))$. As such, this provides an idea of the size and nature of the basins of attractions of optima. Landscapes with high density-basin information should thus be easier for an evolutionary search process to become "attracted" to a fitter solution space and the converse for landscapes with lower density-basin information.

In summary, higher values of information content, partial information content and information stability suggest higher degrees of epistasis and modality, which leads to a more rugged landscape that is harder to search. At the same time, the density-basin information should further assist in determining the search space difficulty by characterizing the landscape around optima. Therefore, using these information-theoretic measures to compare between different artificial evolutionary systems should provide useful characterizations of the search difficulty associated with the different fitness landscapes.

4.4 Experimental Setup

Three series of experiments were performed to provide an insight into the search space difficulty associated with each type of ANN architecture. The fitness of each genotype in these experiments was evaluated according to the f_1 objective function only, which is the locomotion distance achieved by the controller as defined in Section 3.4.1.

In the first series of experiments, random sampling of solutions was conducted for all four architectures. Since random search is used, each genotype is

generated independently from all other genotypes. Furthermore, the variable number of hidden units for each network specified by the genotype is initialized randomly ranging between 0 to 15 hidden units according to a uniform distribution. From prior experiments, it was found that the best controllers only required between 2–4 hidden units (Teo and Abbass 2002a; Teo and Abbass 2002b; Teo and Abbass 2002c). As such, we have chosen to set the upper bound for this parameter at 15 hidden units. In each run, a total of 30,000 genotypes were sampled. This is equivalent to an evolutionary run with a population size of 30 over 1000 generations, which is the intended setup for later experiments involving artificial evolution of ANNs, to ensure that a fair comparison of the search space can be made. Sampling of the search space using random search is replicated for 10 different seeds giving 10 independent runs with a total of 300,000 fitness evaluations (although a single run that directly generates the required 300,000 fitness evaluations would be equivalent, the setup using 10 runs initialized from 10 seeds is used to maintain consistency across all experimental setups).

In the second series of experiments, trial solutions were obtained using a hill-climbing algorithm for all four ANN architectures. New genotypes were generated from the currently accepted genotype using a mutation of 0.1. The mutation operator changes both the values of the connection weights and number of active hidden units in the network. A move from the best solution found so far to a trial solution is accepted only if the trial solution has a higher fitness than the best solution found so far. Otherwise, another trial solution is generated. The fitness for all solutions generated during the hill-climb were recorded and analyzed. Each run was again allowed to sample a maximum of 30,000 genotypes over 10 independent runs as in the random search experiments. In order to reduce the amount of bias on the number of hidden units present in each ANN during initialization of the genotype, each of the 10 independent runs was started using networks initialized with increasing probabilities of having more hidden units ranging from 0 to 15 hidden units. This guarantees that the genotype space is sampled uniformly in terms of the variable hidden layer.

For the last series of experiments, a random walk was performed using all four ANN architectures. The fitness value obtained for every genotype at each step of the walk was recorded. A new point in the search space was generated through a 0.1 mutation of the previous genotype reached during the walk. Again to ensure fair comparisons, each walk was allowed a maximum of 30,000 steps and 10 separate walks were carried out staring from different points in the search space. Also, as with the hill-climbing experiments, each of the 10 independent runs were started using networks initialized with increasing probabilities of having more hidden units to ensure that the genotype space is sampled uniformly in terms of the variable hidden layer.

The information content analysis was carried out only for the search spaces sampled using random walk. This is due to the neighborhood definition of this landscape measure, which is only meaningful when all subsequent fitness points of genotypes sampled in the search space are related through a walk obtained using the genetic operators (see Section 4.1). Furthermore, from the definition of the autocorrelation function, to conduct such an analysis requires sampling of fitness points that are separated by a distance of Γ. However, an autocorrelation analysis was not possible in our particular problem as the algorithm used in our random walk cannot guarantee that fitness points of step length > 1 are unique points in the landscape. This was due to the fact that the mutation operator used to generate subsequent neighborhood points is non-directional in the sense that later mutations may return a particular walk to a previously encountered landscape point. A directional approach would also not provide an accurate picture of the actual fitness step lengths because of two problems: (1) network symmetry, and (2) hidden unit activation. Firstly, mutations arising from distinctly different trajectories can lead to identical ANNs by virtue of architectural symmetries that can arise in the network. Secondly, the flipping of a single bit in the genotype which turns a particular hidden unit on or off will cause an entire set of connection weights to either become active or inactive in the ANN and hence, the mutation of a single hidden unit can cause a very large change in the phenotype. These are known problems in measuring diversity

when evolving neural networks (Yao 1999). As such, an autocorrelation analysis conducted in this case will not be reliable since we cannot guarantee that fitness points after s steps are actually at a distance $\Gamma = s$ away from each other. Hence, we focus our fitness landscape analysis of neighborhood points sampled by random walk using only the informational measures.

This initial investigation of the underlying fitness landscapes should provide some indication on which of the four proposed types of ANN architecture would allow for better controller evolution. Although there are limitations associated with fitness landscape analysis methods, which we discuss in Section 4.6, the results from this initial investigation will at least provide some basis for deciding which type of ANN architecture is to be used for the remainder of the artificial evolution experiments.

4.5 Results and Discussion

The results from the experiments described above are presented in three sections. The first section provides a characterization of the search space using random sampling, followed by hill-climbing and finally using random walk. For each sampling technique, a 3D graph was first plotted to show the frequency distribution of sampled genotypes in terms of their solution fitness as well as the number of hidden units present in the ANN. As the objective is a continuous function, genotypes were grouped into 5000 discrete intervals to calculate the frequency distribution. These 3D frequency graphs were rotated along the X-Y axis in order to provide a clearer depiction of the distribution characteristics as it was found that the default orientation in the original plots often resulted in some features becoming obscured by large peaks in the distribution. This convention is adopted throughout the thesis whenever such graphs are depicted. Additional 2D smoothed contour graphs were plotted to provide a clearer depiction of the relationship between number of hidden units present in the ANN and quality of solutions. A smoothed probability density function was also plotted for each architecture by grouping solutions according to their fitness irrespective of the number of hidden units present in the ANN. Also,

the best solution found by each of these algorithms over the 10 independent runs was plotted over time. This will give an indication of how the search proceeded over time. Finally, a comparison of the best overall solution found using the different algorithms is discussed along with the average and standard deviation of the best solutions found.

4.5.1 Random Search

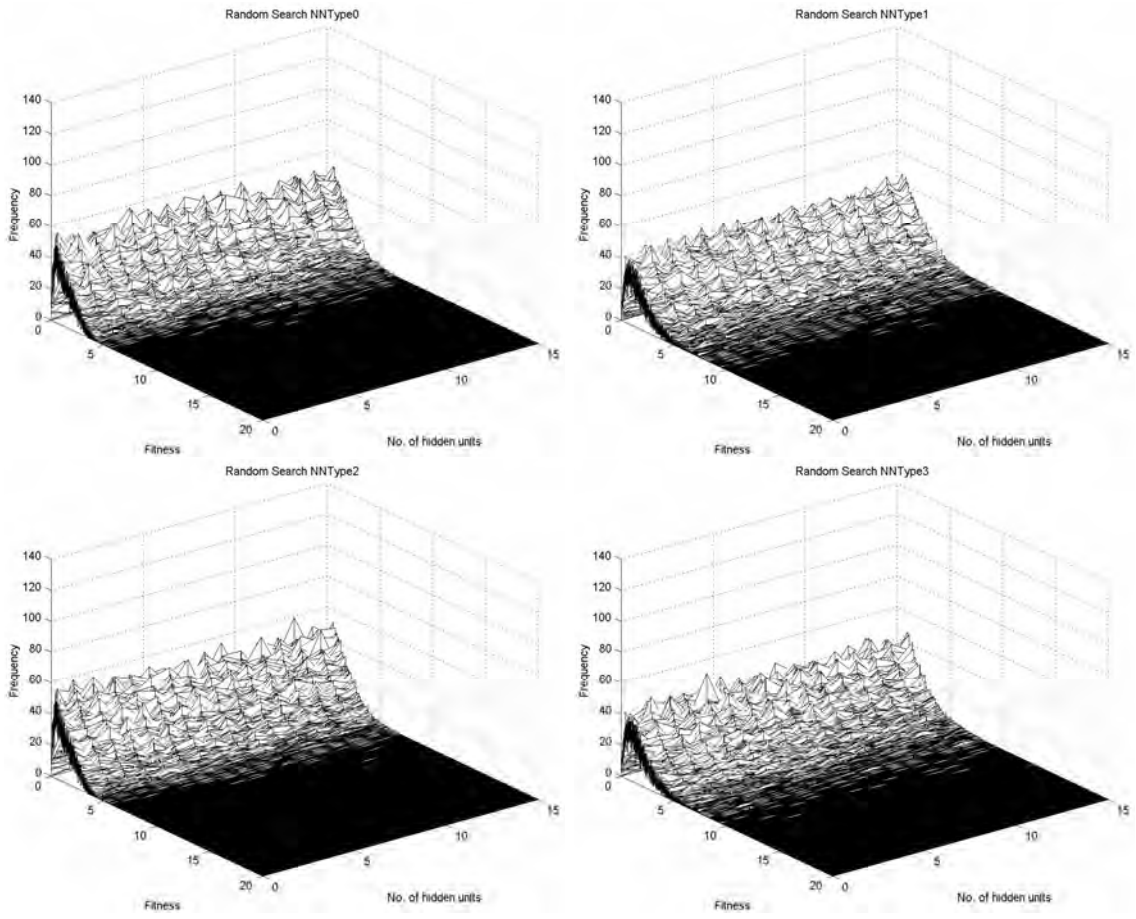


Figure 4.1: Frequency distribution of solutions using random search for ANN architecture 1. NNType0 (top left), 2. NNType1 (top right), 3. NNType2 (bottom left), 4. NNType3 (bottom right). X-axis: Fitness, Y-axis: No. of hidden units, Z-axis: Frequency.

Figure 4.1 is a frequency histogram of the distribution of solutions in terms

of their fitness and number of hidden units. From these graphs, it is apparent that the number of hidden units does not affect the quality of the solutions sampled at random. An analysis conducted on the mean fitness across ANNs with different numbers of hidden units showed that there was no correlation between the number of units present in the hidden layer and the locomotion capability of the ANN. What this means is that from random sampling, controller size in terms of number of hidden units does not appear to affect the locomotion capability of the creature. This may be due to the fact that solutions with good locomotion abilities may be extremely rare and isolated in the search space, which makes this problem of automatic generation of artificial creature controllers a particularly hard problem.

2D contour graphs of frequency distribution of solutions obtained from random search in terms of fitness and number of hidden units present in the ANN are given in Figure 4.2. No obvious concentration of solutions can be seen from these graphs, which confirms the earlier observation that the number of hidden units does not appear to affect the solutions found by random search. However, it is interesting to note that the final contour line extends further by more than 1 unit distance in NNType1 (Figure 4.2.2) and NNType3 (Figure 4.2.4) compared to NNType0 (Figure 4.2.1) and NNType2 (Figure 4.2.3). This suggests that it was slightly easier to reach fitter regions of the controller's objective space using the NNType1 and NNType3 architectures compared to using the NNType0 and NNType2 architectures.

Figure 4.3 shows the probability density function of solutions obtained using random search for all four types of ANN architecture. It is clear from these graphs that a random search of the genotype space yields a very high percentage of low fitness solutions. For all four types of ANN, the most commonly sampled genotype only yields a fitness of around 1. This is a clear indication that a uniform sampling of the genotype space yields a highly skewed distribution of solutions in the objective space. The NNType1 (Figure 4.3.2) and NNType3 (Figure 4.3.4) architectures appear to provide slightly more solutions with higher fitness although the difference is not very obvious. This can be seen from the small shift to the right of the probability curve for these networks. Also, the probability of generating

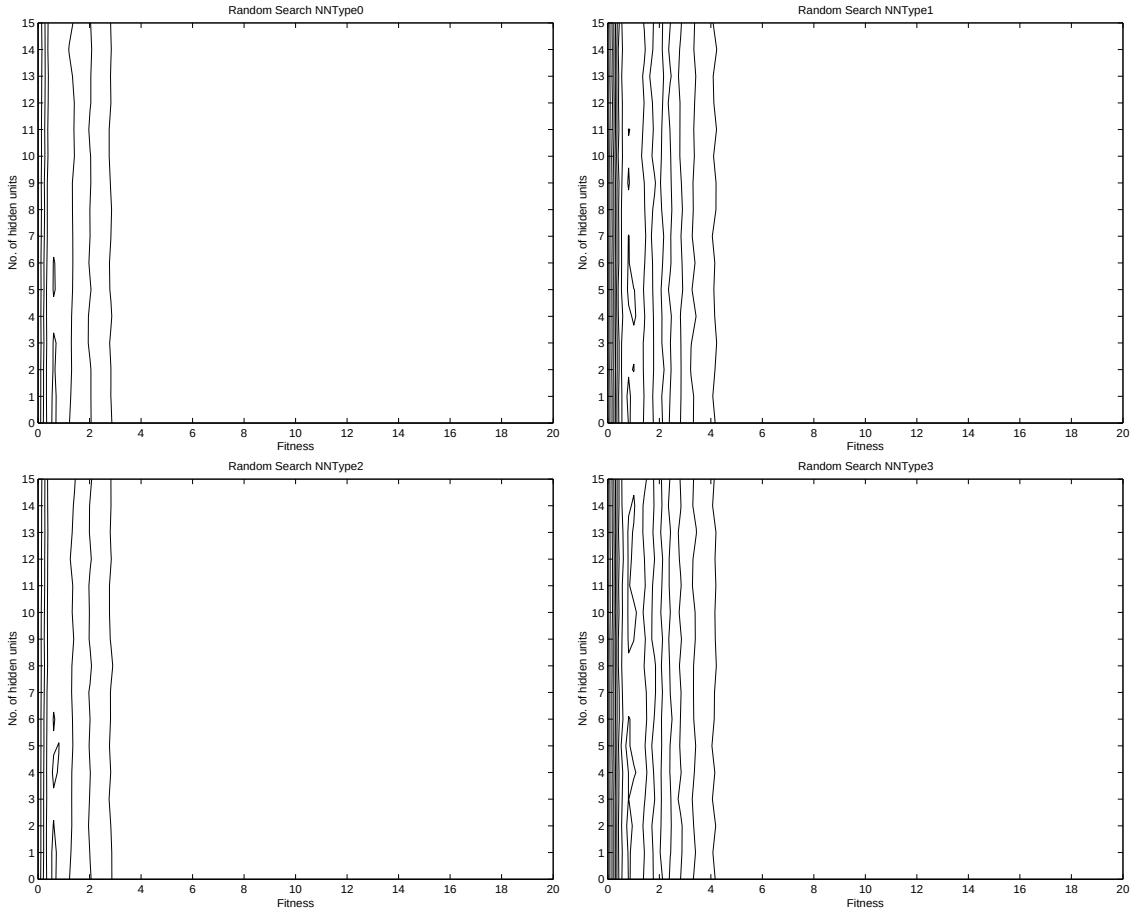


Figure 4.2: Contour graphs of frequency distribution of solutions obtained using random search for ANN architecture 1. NNType0 (top left), 2. NNType1 (top right), 3. NNType2 (bottom left), 4. NNType3 (bottom right). X-axis: Fitness, Y-axis: No. of hidden units.

controllers begins to approach 0 for NNType0 (Figure 4.3.1) and NNType2 (Figure 4.3.3) beyond a fitness 6 whereas for NNType1 and NNType3, this only occurs at a fitness of beyond 8. This observation gives a weak indication that ANNs with direct connections from input to output (NNType1 & NNType3) may be easier to search whereas recurrent connections only (NNType2) do not provide any significant advantage over a standard feed-forward architecture (NNType0).

The best solution obtained over the 30,000 iterations of random search for 10 independent runs is depicted in Figure 4.4. The final best solutions clustered

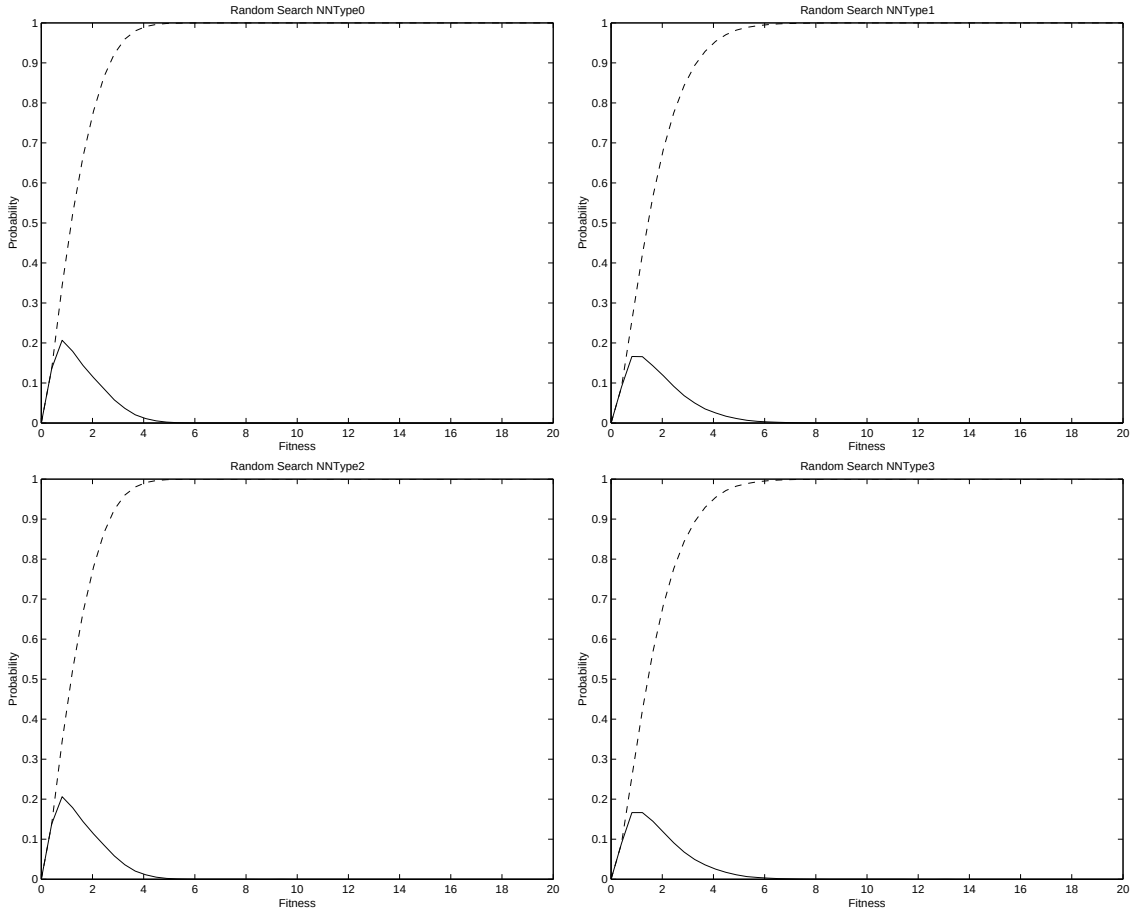


Figure 4.3: Density (solid) and cumulative (dashed) probability distribution of solutions obtained using random search for ANN architecture 1. NNType0 (top left), 2. NNType1 (top right), 3. NNType2 (bottom left), 4. NNType3 (bottom right). X-axis: Fitness, Y-axis: Probability.

between a fitness of 6 to 9 for NNType0 (Figure 4.4.1) and NNType2 (Figure 4.4.3) whereas for NNType1, the final best solutions clustered between 9 and 11 (Figure 4.4.2). There was a larger spread of best final solutions in NNType3 ranging between 9 and 13 (Figure 4.4.4). The NNType3 architecture also had more runs in which significant fitness improvements still occurred in the latter parts of the random search compared to the other three architectures where in most runs, the major improvements in fitness occurred before the 10,000th iteration resulting in large plateau areas in the end regions of these graphs.

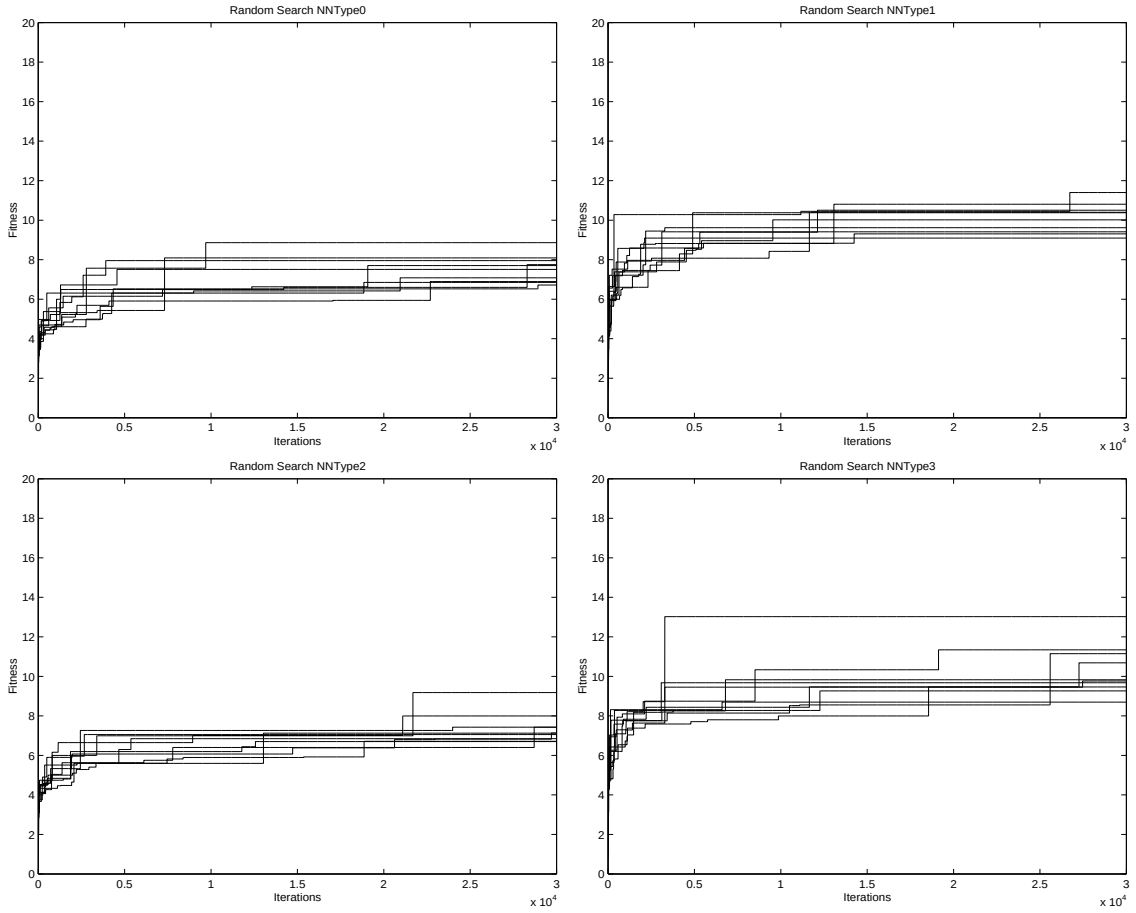


Figure 4.4: Best fitness for solutions obtained over time for 10 runs using random search for ANN architecture 1. NNType0 (top left), 2. NNType1 (top right), 3. NNType2 (bottom left), 4. NNType3 (bottom right). X-axis: Iterations, Y-axis: Fitness.

NNType	Overall Best Fitness	Average Best Fitness ± Standard Deviation	t-statistic (against NNType0)	No. of Hidden Units
0	8.8637	7.5406 ± 0.6733	-	10.0 ± 2.3
1	11.3962	10.0931 ± 0.7348	8.60	7.4 ± 4.0
2	9.1804	7.3609 ± 0.7490	(0.47)	10.9 ± 2.5
3	13.0225	10.2878 ± 1.2747	5.58	9.2 ± 4.2

Table 4.1: Comparison of best solutions found using random search over 10 independent runs.

Table 4.1 shows the overall best f_1 fitness obtained from 10 independent runs of random search along with the average best fitness and standard deviations for all four ANN architecture types. The overall best fitness was obtained using NNType3 followed by NNType1. The next best overall fitness was given by NNType2 and the worst was NNType0. In terms of the average best fitness, NNType3 had the highest value, followed by NNType1, NNType0 and NNType2 respectively. The differences between means of NNType1 and NNType3 against NNType0 were statistically significant at both $\alpha = 0.05$ and $\alpha = 0.01$. This indicates that in terms of the best controllers found, additional input-output connections in NNType1 architectures were able to yield better controllers on average when searched randomly as was the case with NNType3, which had both additional input-output as well as recurrent connections. However, recurrent-only architecture in NNType2 did not show any significant advantages over the standard feed-forward architecture in NNType0.

In terms of the number of hidden units used in the best controllers, the solutions found by random search used an average of between 7.4 and 10.9 hidden units. Surprisingly, the best solutions found using the NNType1 and NNType3 architecture types, which had higher locomotion fitness than NNType0 and NNType2, required on average less number of hidden units compared to these latter two architectures. However, the standard deviations were also much higher in NNType1 and NNType3 suggesting that the apparent inverse relationship between controller size and locomotion distance could have been due to chance encounters with small-sized networks with better locomotion capabilities.

4.5.2 Hill-Climbing

Figure 4.5 plots the frequency distribution of solutions in terms of fitness and number of hidden units. In this case, the number of hidden units does affect the controller's locomotion capabilities as evidenced by the non-uniform distribution of solutions across the objective space. The hill-climbing algorithm appeared to favor genotypes that had between 4 and 10 units in the variable hidden layer as evidenced

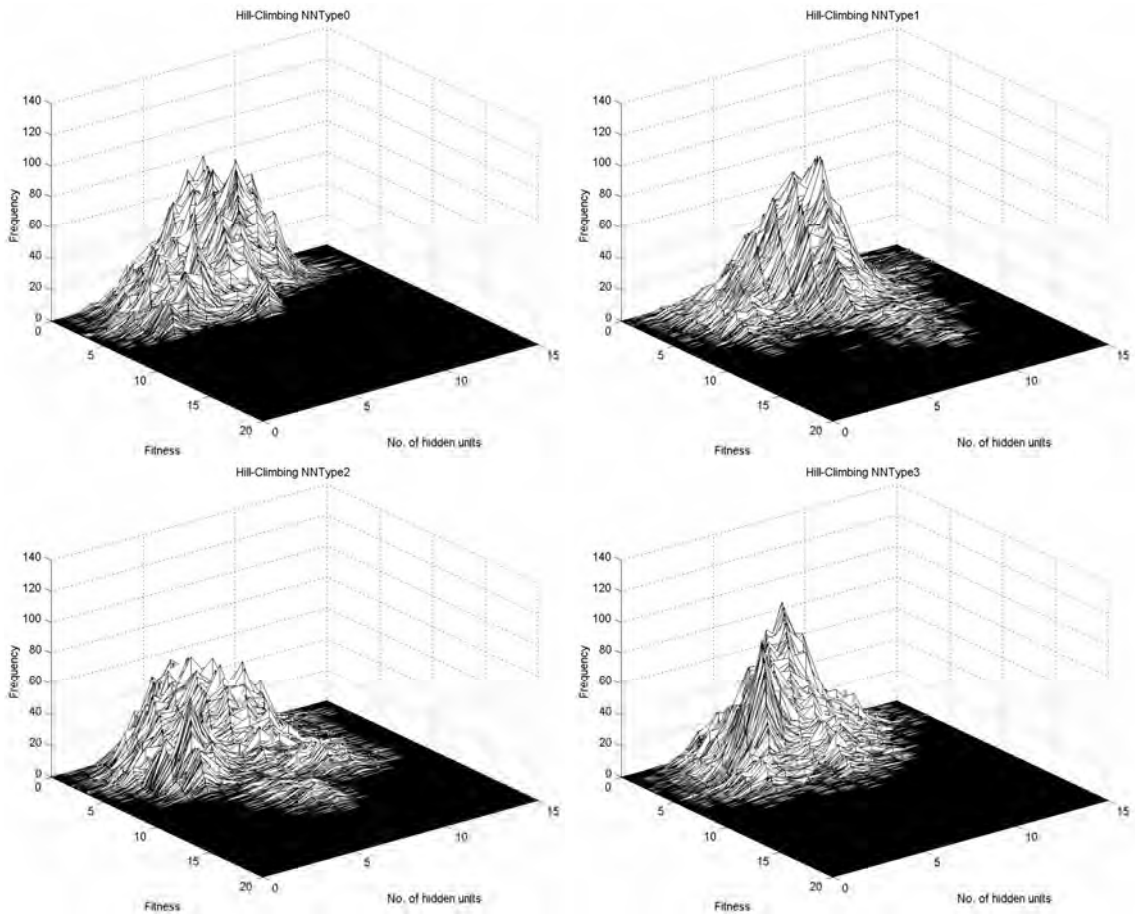


Figure 4.5: Frequency distribution of solutions using hill-climbing for ANN architecture 1. NNType0 (top left), 2. NNType1 (top right), 3. NNType2 (bottom left), 4. NNType3 (bottom right). X-axis: Fitness, Y-axis: No. of hidden units, Z-axis: Frequency.

by the significantly higher frequencies of samples appearing in this region of the objective space.

Accompanying 2D contour graphs of frequency distribution of solutions in terms of fitness and number of hidden units present in the ANN are given in Figure 4.6. The peaks illustrated in these contour graphs provide a clearer picture of where the concentration of sampled genotypes occurred. For NNType0 and NNType1, the most commonly sampled genotypes had hidden layers of 8 units peaking at a fitness of just under 4 (Figure 4.6.1) and 5 (Figure 4.6.2) respectively. NNType2

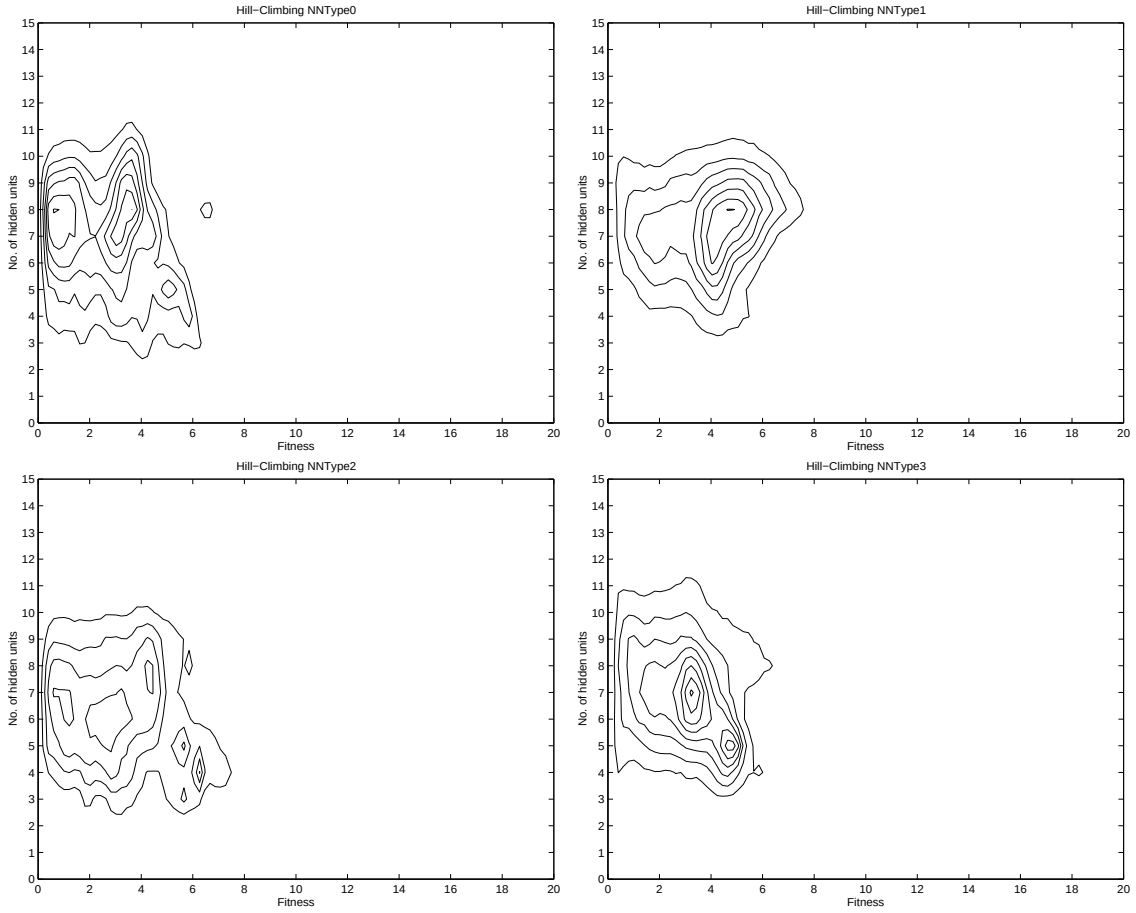


Figure 4.6: Contour graphs of frequency distribution of solutions obtained using hill-climbing for ANN architecture 1. NNType0 (top left), 2. NNType1 (top right), 3. NNType2 (bottom left), 4. NNType3 (bottom right). X-axis: Fitness, Y-axis: No. of hidden units.

had multiple but lower peaks at 4,6,7 and 8 hidden units with fitness ranging from just under 1 to just over 6 (Figure 4.6.3). Finally, the highest concentration of genotypes encountered during hill-climbing in NNType3 had hidden layers of 7 units and fitness of approximately 3.5 (Figure 4.6.4). A slightly lower but still very high peak could also be seen in NNType3 with 5 hidden units and a fitness of around 5. These observations suggest that for all four ANN architectures, hill-climbing is very susceptible to becoming stuck in local optima that are apparently very difficult to break out of.

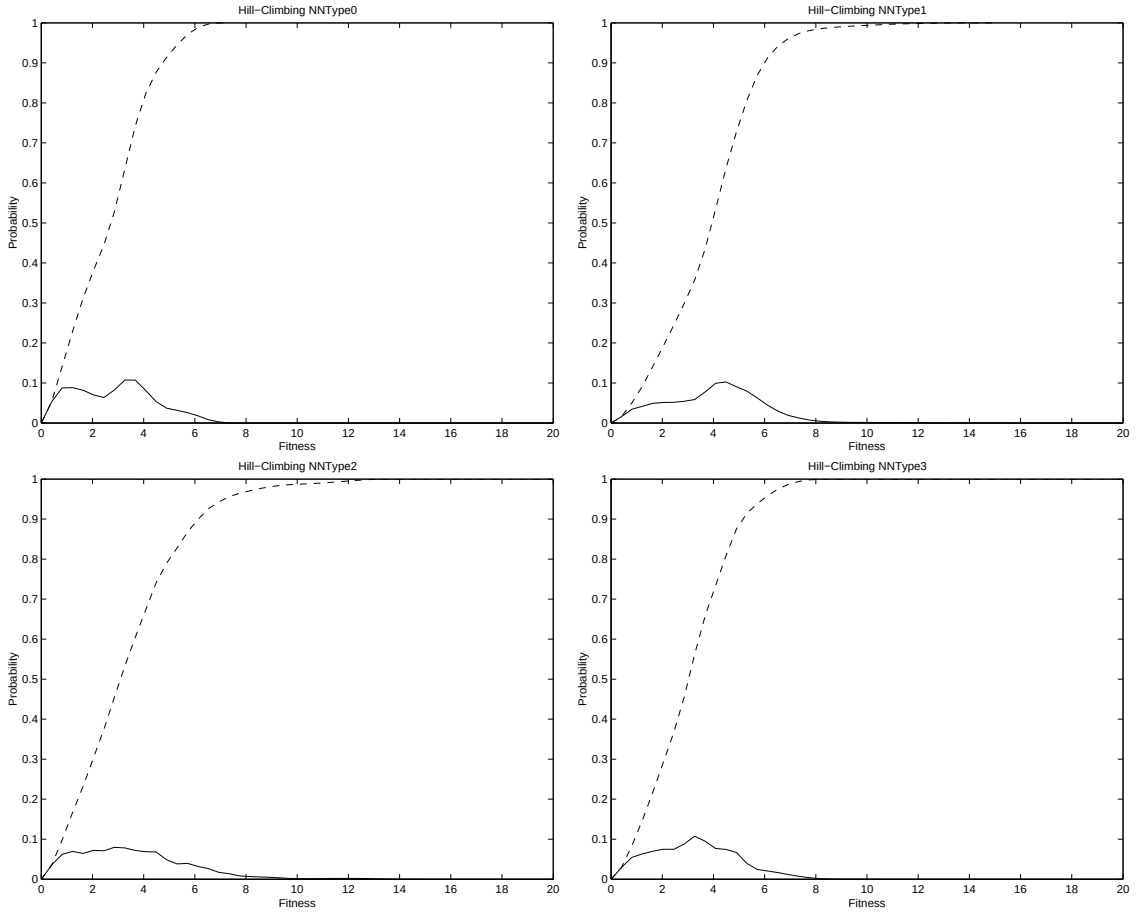


Figure 4.7: Density (solid) and cumulative (dashed) probability distribution of solutions obtained using hill-climbing for ANN architecture 1. NNType0 (top left), 2. NNType1 (top right), 3. NNType2 (bottom left), 4. NNType3 (bottom right). X-axis: Fitness, Y-axis: Probability.

Figure 4.7 shows the probability density function of solutions obtained using hill-climbing for all four types of ANN architecture. The genotypes that were sampled using hill-climbing yielded a set of solutions with much higher fitness than those obtained using random sampling. The architecture that generated the lowest fitness was NNType0 (Figure 4.7.1). Here, the probability of generating a controller approached 0 beyond a fitness of 7. This is expected since the NNType0 architecture has the least number of available connections between layers. Surprisingly, the next best architecture was NNType3 (Figure 4.7.4) which had the most number of

available connections between layers. The probability of generating a controller in this case approached 0 beyond a fitness of only 10. For NNType1 (Figure 4.7.2) and NNType2 (Figure 4.7.3), the probability only approached 0 beyond a much higher fitness of 14. This may be due to a chance encounter with a much fitter solution which caused the sampling process to cluster around a local optimum with a higher fitness, thereby biasing the distribution of solutions towards this area of the objective space (the presence of outliers in NNType1 and NNType2 is discussed again later in this section). It should be noticed though that the majority of the solutions were still sampled around the low quality areas of the search space yielding controllers with locomotion distances of between 0 and 6. This is again an indication that the landscape might be quite rugged and thus very easy for a hill-climbing algorithm to become stuck in a local optimum.

The best solution obtained over the 30,000 iterations of hill-climbing for 10 independent runs is depicted in Figure 4.8. The final solutions appeared to cluster between a fitness of 4 to 6 and is most apparent in NNType0 (Figure 4.8.1). The best solutions obtained with NNType1 (Figure 4.8.2) and NNType2 (Figure 4.8.3) had a larger spread of fitness values compared NNType3 (Figure 4.8.4). What is noticeably clear is that most of the improvement in the quality of solutions occurred within an extremely short window at the start of the search process and subsequent improvements were minimal except only in a single run each with NNType1 and NNType2 causing large plateau areas in the graphs. This supports the earlier hypothesis that the landscape may be quite rugged and that a hill-climbing algorithm may get stuck very easily in a local optimum and find it difficult to obtain fitter solutions that will enable it to move away from the local optimum.

Table 4.2 shows the overall best f_1 fitness obtained from 10 independent runs of hill-climbing along with the average best fitness and standard deviations for all four ANN architecture types. The overall best fitness was obtained using NNType1 although the overall best fitness from NNType2 was only less by 0.47. This was followed by NNType3 and the worst was NNType0. This is consistent with the mean of the best fitness which also indicates that NNType1 and NNType2

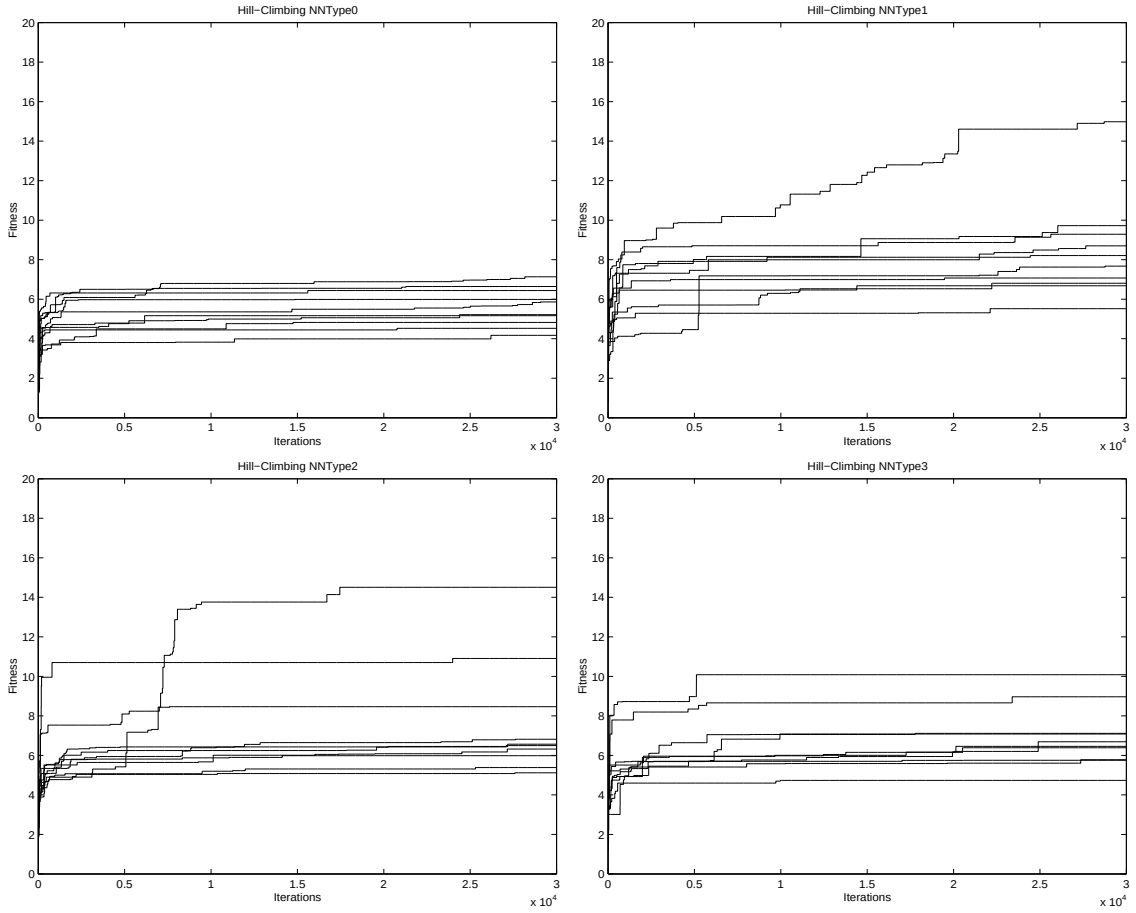


Figure 4.8: Best fitness for solutions obtained over time for 10 runs using hill-climbing for ANN architecture 1. NNType0 (top left), 2. NNType1 (top right), 3. NNType2 (bottom left), 4. NNType3 (bottom right). X-axis: Iterations, Y-axis: Fitness.

NNType	Overall Best Fitness	Average Best Fitness ± Standard Deviation	t-statistic (against NNType0)	No. of Hidden Units
0	7.1333	5.5969 ± 0.9714	-	7.0 ± 2.3
1	14.9792	8.4652 ± 2.6246	3.41	7.4 ± 2.0
2	14.5086	7.6568 ± 2.9365	2.06	6.4 ± 2.5
3	10.0832	6.9057 ± 1.5719	2.18	6.8 ± 2.2

Table 4.2: Comparison of best solutions found using hill-climbing over 10 independent runs.

were the easiest architectures to search using hill-climbing in generating efficient controllers for the creature. Correspondingly, the worst architecture was NNType0 which had the least possible number of connections allowable between layers of the network. However, it should be noted that the standard deviations for NNType1 and NNType2 were higher than NNType3 or NNType0 and may indicate the presence of outliers that were chanced upon during the search. A t-test showed that the only significant difference was between NNType0 and NNType1. NNType2 and NNType3 did not show any significant differences in terms of their average best solution compared to NNType0 at both $\alpha = 0.05$ and $\alpha = 0.01$. This is an indication that in terms of the best solutions found using a hill-climbing algorithm, only additional input-output connections (NNType1) were advantageous in yielding higher quality locomotion controllers and that neither additional recurrent-only (NNType2) nor additional recurrent plus input-output connections (NNType3) provided any significant advantages over the standard feed-forward architecture (NNType0).

There was very little difference in the number of hidden units used by the best controllers found using hill-climbing. On average, the best solutions found using NNType2 required the least number of hidden units at 6.4 while NNType1 required the most at 7.4. The standard deviation among the best controllers was also very similar across the different architectures ranging between 2.0 and 2.5 hidden units.

4.5.3 Random Walk

The frequency distribution of solutions obtained from a random walk of the fitness landscape is presented in Figure 4.9. All four architectures yielded a fairly similar but again highly skewed distribution over the objective space. The majority of genotypes sampled by a random walk again clustered around the very low quality areas of the search space and around ANNs with hidden layers of between 3 and 12 units. As such, a very high percentage of low fitness solutions again appeared to dominate the random walk. As with hill-climbing, there are indications that the size of the hidden layer affects the locomotion capabilities of the controller. This is more evident from the contour graphs that follow.

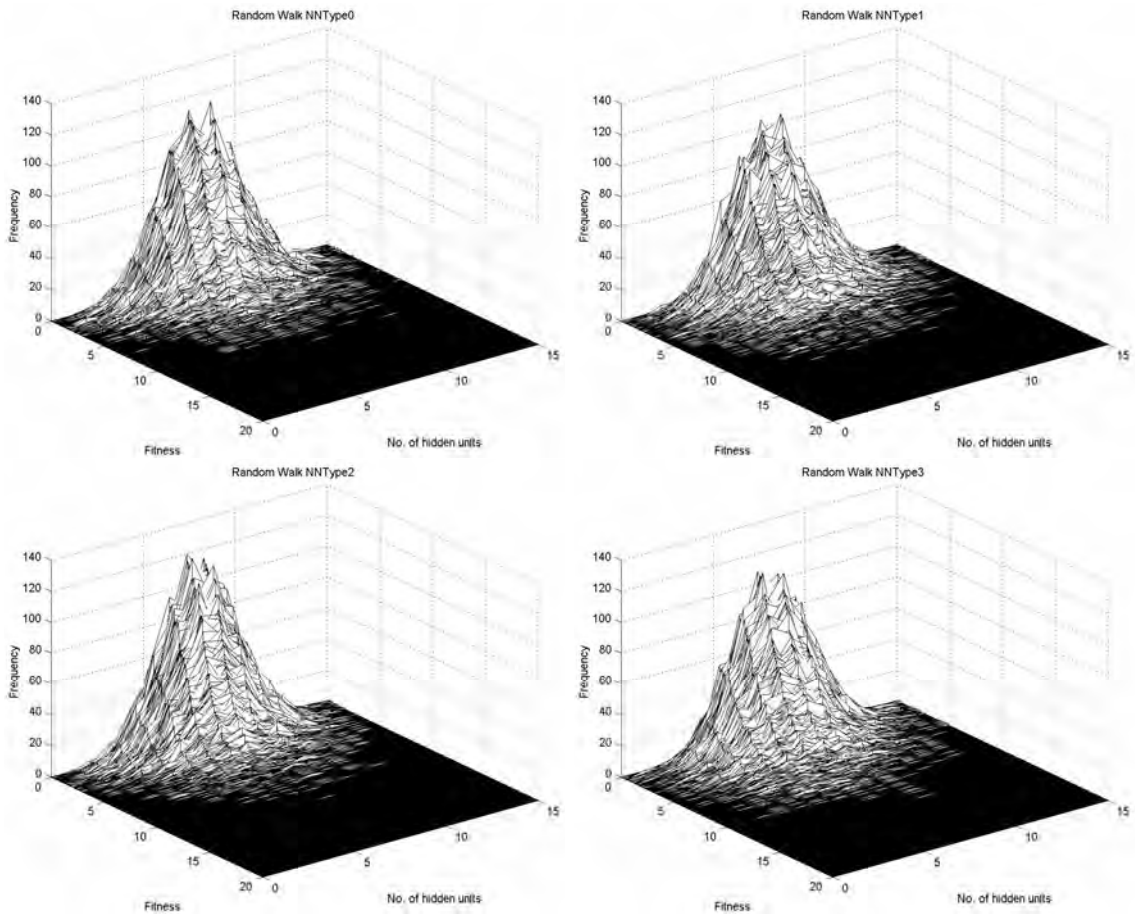


Figure 4.9: Frequency distribution of solutions using random walk for ANN architecture 1. NNType0 (top left), 2. NNType1 (top right), 3. NNType2 (bottom left), 4. NNType3 (bottom right). X-axis: Fitness, Y-axis: No. of hidden units, Z-axis: Frequency.

The effect of hidden units on the fitness of genotypes is very apparent in these accompanying 2D contour graphs depicted in Figure 4.10. Solutions with fitness above 4 had between 6 and 9 hidden units. It is also clear from the peaks on these graphs that the most frequently encountered genotype had between 7 and 8 hidden units. This may indicate that there is a large basin of attraction in this region of the search space. However, the fitness of solutions in this area is very low indeed (~ 1). Similar to the observations noted in the random search contour graphs, it was slightly easier to reach fitter regions of the controller's objective

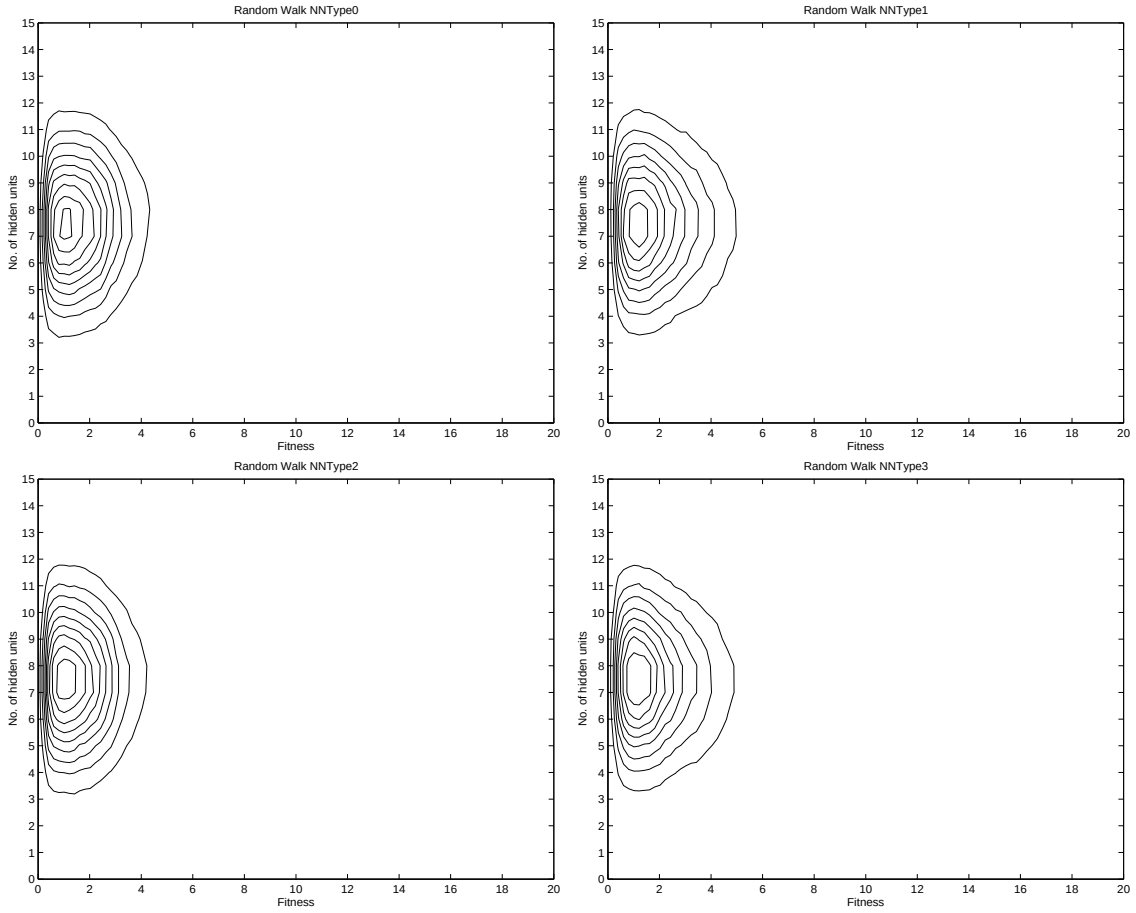


Figure 4.10: Contour graphs of frequency distribution of solutions obtained using random walk for ANN architecture 1. NNType0 (top left), 2. NNType1 (top right), 3. NNType2 (bottom left), 4. NNType3 (bottom right). X-axis: Fitness, Y-axis: No. of hidden units.

space when random walks were performed on the fitness landscape of ANNs with architecture NNType1 (Figure 4.10.2) and NNType3 (Figure 4.10.4) compared to NNType0 (Figure 4.10.1) and NNType2 (Figure 4.10.3). However, this effect is not very significant and thus only gives a weak indication that ANNs with direct connections from input to output (NNType1 & NNType3) may be easier to search than those with recurrent connections only (NNType2) or a standard feed-forward architecture (NNType0).

The probability density function of solutions obtained using random walk

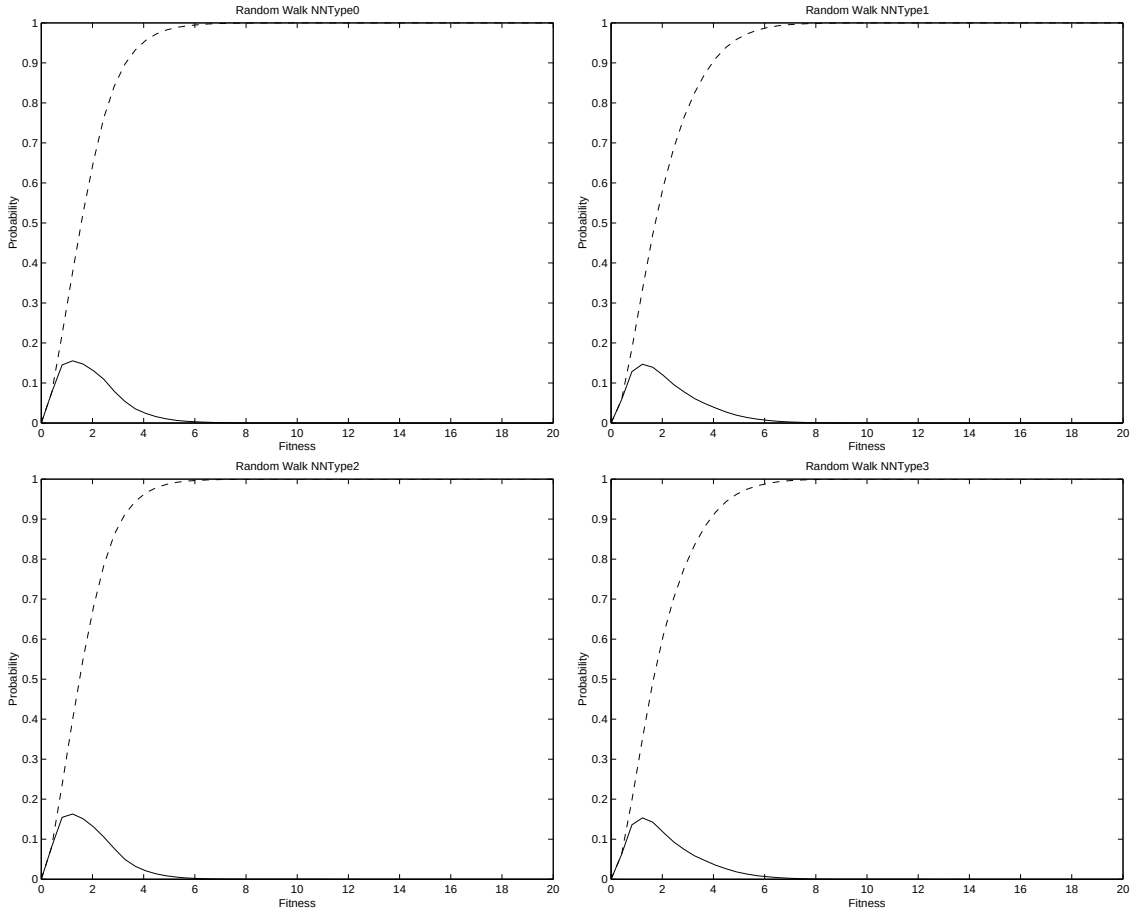


Figure 4.11: Density (solid) and cumulative (dashed) probability distribution of solutions obtained using random walk for ANN architecture 1. NNType0 (top left), 2. NNType1 (top right), 3. NNType2 (bottom left), 4. NNType3 (bottom right). X-axis: Fitness, Y-axis: Probability.

is illustrated in Figure 4.11 for all four ANN architectures. The shape of the curves was very similar to the ones obtained with random search. These graphs show that a random walk of the genotype space yields a very high percentage of low fitness solutions centered around a fitness of only 1. This supports the earlier observation from random search that the distribution of solutions in the objective space is highly non-uniform. As with random search, random walk had a slightly better probability of encountering fitter genotypes with NNType1 and NNType3 architectures compared to the NNType0 and NNType2 architectures (as evidenced by the slightly

larger areas under the curves in Figures 4.11.2 & 4.11.4 compared to the curves in Figures 4.11.1 & 4.11.3) as the probabilities approached 0.

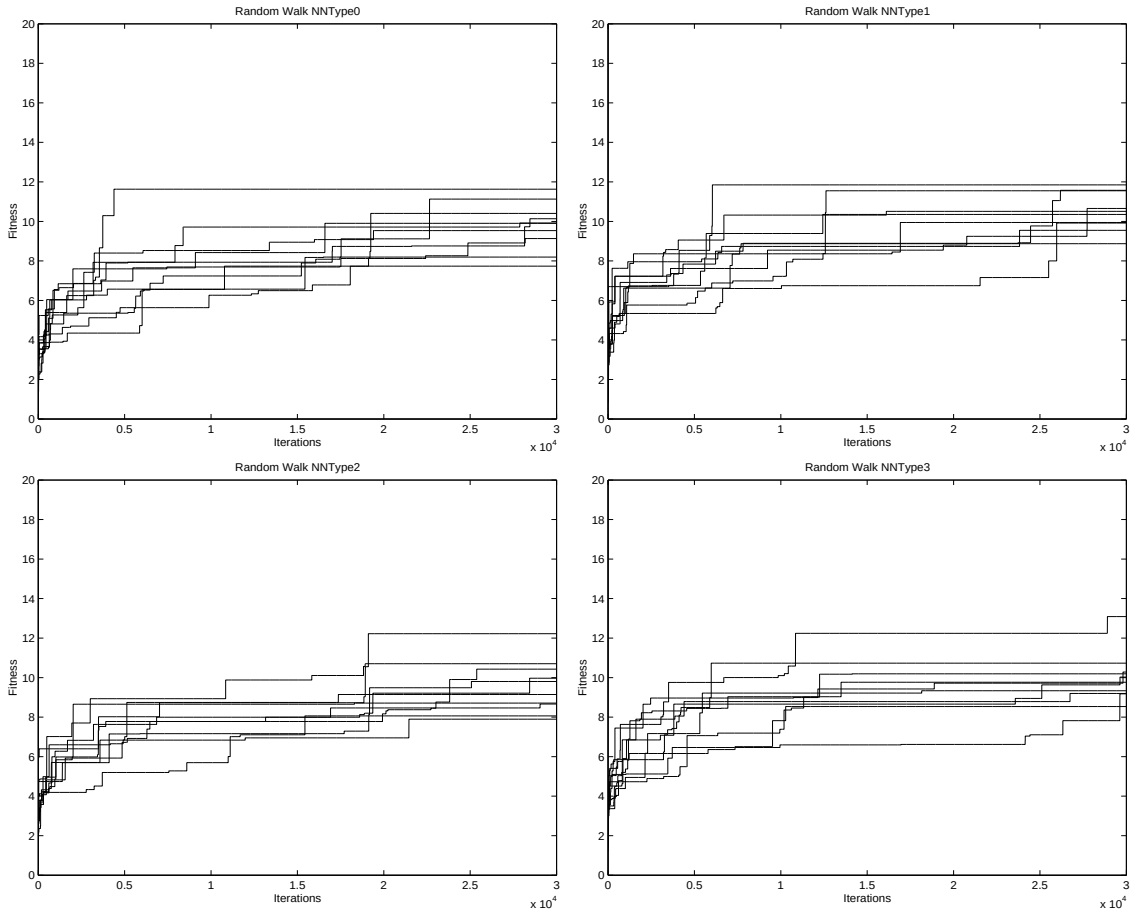


Figure 4.12: Best fitness for solutions obtained over time for 10 runs using random walk for ANN architecture 1. NNType0 (top left), 2. NNType1 (top right), 3. NNType2 (bottom left), 4. NNType3 (bottom right). X-axis: Iterations, Y-axis: Fitness.

The best solution obtained over the 30,000 iterations of random walk for the 10 independent runs is depicted in Figure 4.12. Compared to hill-climbing, the best solutions obtained for all four ANN architectures were less clustered within a specific range of fitness and had a fair spread of solutions between 7 and 12. Again this supports the earlier observations that the fitness landscape may be quite rugged and thus in a random walk, which does not have the constraint of having

to search within a neighborhood area of the best solution found so far, may have a better chance of finding a better solution by virtue of its random trajectory through different objective sub-spaces. Additionally, the progression of the best solution over time is more gradual in random walk compared to hill-climbing and the periods where the best fitness does not improve is also much shorter as evidenced by less occurrences of long plateau regions.

NNType	Overall Best Fitness	Average Best Fitness ± Standard Deviation	t-statistic (against NNType0)	No. of Hidden Units
0	11.6325	9.7725 ± 1.2009	-	7.4 ± 2.1
1	11.8494	10.4776 ± 0.9616	1.66	7.4 ± 2.2
2	12.2220	9.5602 ± 1.3372	(0.43)	8.2 ± 2.0
3	13.0900	10.0333 ± 1.2535	0.42	7.0 ± 1.4

Table 4.3: Comparison of best solutions found using random walk over 10 independent runs.

Table 4.3 shows the overall best f_1 fitness obtained from the 10 independent runs of random walk along with the average best fitness and standard deviations for all four ANN architecture types. A slightly different picture is given by random walk compared to hill-climbing in terms of the ease of searching for good controllers across the four ANN architectures. The results obtained were less differentiating for the overall best fitness and especially with the average of the best fitness. Here, although the overall best fitness was highest for NNType3 followed by NNType2, then by NNType1 and finally NNType0, the averages had NNType1 with the highest best fitness followed by NNType3, then by NNType0 and the worst was NNType2. Taking into consideration the standard deviations, the means of these best solutions were not very different from each other. As such, there are no strong indications as to what effect allowing recurrency and direct input-output connections have on the ease of searching for good quality controllers. It is also interesting to note that a comparison of the average best fitness obtained with random walk was much higher than those obtained with hill-climbing. This lends further confirmation to the fact that hill-climbing is highly inefficient in searching this landscape and that even a random walk is better in chancing upon a fitter solution. A t-test at both

$\alpha = 0.05$ and $\alpha = 0.01$ showed no significant differences for the four different ANN architectures in terms of the best solutions obtained over 10 independent runs.

As with hill-climbing, there was little variation in terms of the size of the hidden layer among the best controllers found using random walk. However in this case, NNType3 required on average the least number of hidden units, NNType0 and NNType1 had similar requirements while NNType2 required the highest number of hidden units. Also, on average across all architecture types, random walk used approximately 1 hidden unit more than the best controllers found using hill-climbing and approximately 2 hidden units less than random search.

4.5.3.1 Information Content Analysis

NNType	ϵ	$H(\epsilon)$	$M(\epsilon)$	Exp. No. of Optima	$h(\epsilon)$
0	0	0.4067 ± 0.0009	0.6226 ± 0.0050	9339 ± 75	0.5727 ± 0.0030
	1	0.6733 ± 0.0269	0.2406 ± 0.0219	3608 ± 328	0.3993 ± 0.0293
	2	0.3442 ± 0.0522	0.0771 ± 0.0186	1156 ± 279	0.1588 ± 0.0310
	5	0.0233 ± 0.0135	0.0023 ± 0.0015	34 ± 22	0.0059 ± 0.0039
	9	0.0002 ± 0.0004	0.0000 ± 0.0000	0 ± 0	0.0000 ± 0.0001
	12	0.0000 ± 0.0000	0.0000 ± 0.0000	0 ± 0	0.0000 ± 0.0000
1	0	0.4066 ± 0.0005	0.6302 ± 0.0042	9452 ± 64	0.5681 ± 0.0026
	1	0.6951 ± 0.0358	0.2581 ± 0.0275	3870 ± 413	0.4266 ± 0.0309
	2	0.3837 ± 0.0539	0.0914 ± 0.0226	1371 ± 339	0.1856 ± 0.0354
	5	0.0338 ± 0.0170	0.0031 ± 0.0019	47 ± 29	0.0091 ± 0.0054
	9	0.0003 ± 0.0004	0.0000 ± 0.0000	0 ± 0	0.0000 ± 0.0001
	12	0.0000 ± 0.0000	0.0000 ± 0.0000	0 ± 0	0.0000 ± 0.0000
2	0	0.4069 ± 0.0007	0.6241 ± 0.0042	9361 ± 63	0.5718 ± 0.0025
	1	0.6703 ± 0.0220	0.2379 ± 0.0169	3568 ± 254	0.3924 ± 0.0236
	2	0.3327 ± 0.0418	0.0720 ± 0.0144	1079 ± 215	0.1514 ± 0.0248
	5	0.0186 ± 0.0079	0.0017 ± 0.0008	26 ± 12	0.0045 ± 0.0020
	9	0.0003 ± 0.0004	0.0000 ± 0.0000	0 ± 0	0.0000 ± 0.0001
	12	0.0000 ± 0.0000	0.0000 ± 0.0000	0 ± 0	0.0000 ± 0.0000
3	0	0.4067 ± 0.0005	0.6303 ± 0.0037	9454 ± 55	0.5680 ± 0.0023
	1	0.6976 ± 0.0365	0.2608 ± 0.0281	3912 ± 421	0.4267 ± 0.0306
	2	0.3817 ± 0.0548	0.0911 ± 0.0218	1366 ± 327	0.1845 ± 0.0366
	5	0.0330 ± 0.0189	0.0032 ± 0.0020	48 ± 30	0.0090 ± 0.0060
	9	0.0002 ± 0.0004	0.0000 ± 0.0000	0 ± 0	0.0000 ± 0.0001
	12	0.0000 ± 0.0000	0.0000 ± 0.0000	0 ± 0	0.0000 ± 0.0000

Table 4.4: Information content analysis using random walk for the 4 ANN architecture types.

The information characteristics associated with the search spaces of the four different types of controller architectures are presented in Table 4.4. This analysis does not indicate any discernable differences between the four different fitness landscapes in terms of information content. All architectures have a similarly high information stability ($H(\epsilon) = 0$) of approximately 12 indicating that the differences in fitness between neighboring solutions is very high. $H(0)$ is also quite large and therefore this indicates that the diversity of shapes on the landscape is also relatively high. $M(0)$ is also large indicating that a high degree of modality was encountered during the walk, which is also evident from the large number of expected optima on the landscape. Surprisingly, $h(0)$ is significantly large, indicating that there are diverse flat and smooth landscape sections. As such, the information content analysis points to the fact that there is a mixture of both rugged as well as smooth areas in the fitness landscapes for all four architectures, the characteristics of which were both encountered a significant proportion of the time during the random walk. This is an indication that depending on which sub-spaces were being explored, the characteristics of the landscape may differ very substantially from highly rugged to very smooth. As such, the ability for search algorithms to find increasingly better solutions may be highly dependent on the initialization and trajectory of the search on the fitness landscape.

4.6 Limitations and Future Work

The idea of neutral plateaus within search spaces, where large numbers of genotypes have similar phenotype fitness values, has recently been of particular interest (Huynen 1996; Barnett 1998; Smith, Philippides, Husbands, and O'Shea 2002). It was suggested that problems with high degrees of neutrality, whether an inherent feature of the original problem or artificially introduced through genotype redundancy, tend to produce landscapes that are easier for EAs to escape local optima and find better solutions (Shackleton, Shipman, and Ebner 2000; Vassilev and Miller 2000). However, both autocorrelation and information content measures

are unable to provide any information regarding the presence of such neutral areas within search spaces (Barnett 1998; Smith, Philippides, Husbands, and O'Shea 2002). A new methodology for elucidating neutrality was proposed by Smith, Husbands, Layzell, and O'Shea (2002). However, these methods measure for evolvability rather than providing direct characterizations of the actual fitness landscapes. Furthermore, there is also evidence that neutrality does not necessarily improve the evolutionary search process (Smith, Husbands, and O'Shea 2001a) and hence, the general significance of neutrality within search spaces remains somewhat inconclusive.

In order for both autocorrelation and information content measures to work, an important assumption needs to be made — that the fitness landscapes being analyzed are statistically isotropic (Weinberger 1990; Vassilev, Fogarty, and Miller 2000). Importantly, it was highlighted by Smith, Husbands, and O'Shea (2001b) that the search space for an evolutionary robotics task environment displayed strong indications of anisotropy. Therefore, the results obtained from landscape analysis methods that make the explicit assumption of isotropy needs to be treated with some caution. Another important observation made by Smith, Husbands, and O'Shea (2001b) is that results obtained from using sparse sampling methods such as random sampling and random walk may provide a highly inaccurate picture of the actual fitness landscape when the distribution of solutions in the search space is non-normal and highly skewed, as was the case in the experiments carried out in this study. This problem of heterogeneity is present in many hard problems and may lead to inaccurate characterizations of landscapes when using techniques that assume homogenous distribution of solutions. Another landscape feature that may also affect the efficacy of the evolutionary search process is the degree of deceptiveness of the problem and again is not a characteristic that can be ascertained with current landscape measures (Smith, Husbands, and O'Shea 2001b).

From our experiments, we have also noted five additional limitations associated with these existing landscape analysis methods. Where relevant, we provide some pointers on open research questions and possible future work that will further

extend the usability and generalization power of these techniques.

1. The problems being analyzed are generally problems that exist in high-dimensional space whereas the landscape analysis methods such as the autocorrelation and information content measures only provide some statistical characterization of the actual search space. As such, these methods only capture a limited amount of information along a particular dimension. A methodology that is able to capture more information from the high-dimensional search spaces would thus be desirable in order to give a more comprehensive and accurate appraisal of the actual fitness landscape.

2. Obtaining the landscape points through a random walk results in evaluating the search space in one particular direction. This implies a bias in the way in which the landscape is being characterized. In order to reduce this bias, multiple walks need to be carried out. However, generating multiple random walks is extremely time-consuming and the time spent on characterizing the fitness landscape may actually take longer than that needed to simply proceed with the actual optimization process of finding a solution. More research effort is required towards designing a computationally more efficient technique for obtaining fitness values in fitness landscape analyses.

3. The operators involved in generating landscape points function only in the genotype space. As such, these landscape measures do not provide any information whatsoever concerning the genotype-to-phenotype mapping. Again, it would be highly desirable to have a technique that is able to give some insights into how different genotype-phenotype mappings affect the fitness landscapes.

4. Current landscape analysis methods only work with a single fitness function. If the problem is multi-objective, then none of the existing measures are able to generalize to higher dimensional objective spaces. For example, the parameter ϵ from the information content analysis can only be used to perform analysis on one objective function at a time. Furthermore, it does not show the degree of correlation between the objective functions. With the resurgent interest

in multi-objective optimization approaches for solving real-world problems, an analysis technique able to characterize such multi-functioned landscapes would be of great value to both researchers and industry practitioners alike.

5. The last and perhaps most serious drawback to current landscape analysis techniques is their inability to capture the true landscape of evolving populations in EAs. Current methods rely on generating a single genotype and tracing its single path through the fitness landscape. It must be remembered that in EAs, an entire population is moving through the fitness landscape, not just a single individual. Considerations need to be given to the coverage of the search space achieved by the evolving population as the optimization progresses. Additionally, concurrent evolutionary paths somehow need to be tracked in order to provide a more accurate picture of how the actual formations present on the EA landscapes affect the transition of populations from one generation to the next.

4.7 Chapter Summary

An analysis on the fitness landscape for four different types of ANN architecture yielded the following results:

- The advantages or disadvantages of having recurrent connections and/or direct input-output connections in the ANN for controlling the artificial creature remain unclear. In terms of average best solutions found, random search performed better using NNType1 and NNType3, hill-climbing performed better using only NNType1 whereas random walk showed no performance differences whatsoever between the four types of architectures. Furthermore, hill-climbing performed worse than both random search and random walk in three out of the four architectures and worse than random walk in the remaining case. As such, whether the search space difficulty is lowered by adding recurrent and/or input-output connections to a standard feed-forward ANN architecture cannot be concluded with certainty.

- The fitness landscape of all four ANN architectures is highly similar. The landscape analysis conducted using informational measures did not show any discernable differences between the four search spaces.

- The fitness landscape of these evolutionary search spaces has both rugged and smooth sections depending on the sub-spaces being explored. Additionally, the variety of rugged shapes on the landscape is high indicating that epistatic interactions between genes in the genotype are high. A correspondingly high degree of modality in the fitness landscape was also noted.

- The solution space is highly heterogeneous — a uniform sampling of the genotype space yielded a highly skewed distribution of solutions in the objective space.

- There are serious deficiencies associated with current landscape analysis methodologies, especially for analyzing non-homogenous and anisotropic search spaces, such as in artificial creature evolution. Additional limitations were also noted for characterizing evolutionary search spaces using such techniques.

It remains unclear as to whether the NNType0, NNType1, NNType2 or NNType3 architecture provides a more amenable search space. As such, experimentation on all four architectures will be required in searching for fit artificial creature controllers. In the next chapter, we will present our evolutionary optimization algorithm using a Pareto multi-objective methodology for evolving controllers based on these four ANN architectures.

Chapter 5

Multi-Objective Controller Evolution

[1] The artificial evolution conducted in prior studies have mainly focused on single objectives, for example walking, swimming, light-following, block-pushing or obstacle avoidance (Sims 1994b; Komosinski and Rotaru-Varga 2000; Hornby and Pollack 2001a; Bongard 2002a). Although there have been some studies that appear to have multi-objectivity present in the evolutionary system, such as predator-prey simulations (Cliff and Miller 1996; Nolfi and Floreano 2000; Floreano, Nolfi, and Mondada 2001), body-brain co-evolution (Bongard and Paul 2000; Hornby and Pollack 2001a) and evolution by physical competition (Sims 1994b), they do not explicitly impose the evolutionary search on distinctly different optimization criteria. In other words, these studies do not explicitly qualify the solutions in terms of a Pareto set (explained in next section), which is a focal concept in evolutionary multi-objective optimization (EMO). Consequently, the resulting artificial creatures cannot exhibit clear trade-offs in terms of their different evolutionary goals. Here, we propose a methodology for multi-objective evolution of creature controllers that emphasizes the generation of Pareto optimal sets of solutions, that is the generation of results that explicitly trade-off between two different and conflicting optimization

[1] Some of the material presented in this chapter have been previously published in Teo and Abbass (2002a; 2002b; 2002c).

objectives. More specifically, we will attempt to simultaneously minimize the number of hidden units used in the creature's ANN controller while at the same time maximize the horizontal distance travelled by the creature as guided by its ANN controller. Hence, our proposed approach will produce a Pareto optimal set of controllers that have clear delineations between optimizing network size and locomotion capability through an EMO process.

First, we explain the concept of non-dominance and Pareto optimality. Then, we present an overview of the literature concerning EMO algorithms. This is followed by a review of the PDE family of EMO algorithms, which our proposed Pareto EMO algorithm for generating controllers is based upon. Next, we explain in detail our proposed algorithm called SPANN. This is then followed by a discussion of the experimental setup for evolving locomotion controllers using the proposed Pareto EMO methodology. As the experiments from the previous chapter did not show any discernable differences between the search space difficulties associated with the four different types of ANN architectures proposed in Section 3.3.3, we will continue to experiment with all four ANN types. This will ascertain whether or not significant advantages can be offered by the different ANN architectures under an EMO paradigm. The remainder of the chapter presents a detailed discussion of the results from these experiments.

5.1 Dominance and Pareto Optimality

The optimization problem (hereafter referred to as P1) can be stated as

$$\text{(P1): Minimise } f(x)$$
$$\text{subject to: } \theta(x) = \{x \in R^n \mid G(x) \leq 0\}$$

where x is the set of decision variables, $f(x)$ is the objective function, $G(x)$ is a set of constraints, and $\theta(x)$ is the set of feasible solutions. If the optimization problem is maximization, it is equivalent to a minimization problem by multiplying the objective by (-1). Also, if a constraint is an equation, it can be represented by two inequalities — one is "less than or equal" and the other is "greater than or

equal". A "greater than or equal" inequality can be transformed to a "less than or equal" inequality by multiplying both sides by (-1). In short, any optimization problem can be represented in the previous general form.

Two important types of optimal solutions will be used in this thesis, local and global optimal solutions. Let us define the open ball, that is a neighborhood centered on $\bar{x}$ and defined by the Euclidean distance δ, as follows

$$B_\delta(\bar{x}) = \{x \in R^n \mid ||x - \bar{x}|| < \delta\}$$

Definition 1: Local optimality A point $\bar{x} \in \theta(x)$ is said to be a local minimum of the optimisation problem **iff** $\exists\ \delta > 0$ such that $f(\bar{x}) \leq f(x),\ \forall x \in (B_\delta(\bar{x}) \cap \theta(x))$.

Definition 2: Global optimality A point $\bar{x} \in \theta(x)$ is said to be a global minimum of the optimization problem **iff** $f(\bar{x}) \leq f(x),\ \forall x \in \theta(x)$.

Usually, there is more than a single objective to be optimized in real life applications. In this case, the problem is called a *multi-objective optimization problem* (MOP). The problem P1 can be re-defined as a general multi-objective optimization problem, MOP1, by replacing the objective function $f(x)$ with a vector of objectives $F(x)$ as follows

$$\text{(MOP1): Minimize } F(x)$$
$$\text{subject to: } \theta(x) = \{x \in R^n \mid G(x) \leq 0\}$$

When the objectives are in conflict, the existence of a unique optimal solution is no longer a valid concept. The solution which satisfies the optimality conditions of one objective may be a bad solution for another. Consequently, we need to redefine the concepts of local and global optimality in multi-objective problems. To do this, we define two operators, $\ncong$ and $\prec$ and then assume two vectors, X and Y. $X \ncong Y$ **iff** $\exists\ x_i \in X$ and $y_i \in Y$ such that $x_i \neq y_i$. $X \prec Y$ **iff** $\forall\ x_i \in X$ and $y_i \in Y,\ x_i \leq y_i$, and $X \ncong Y$. $\ncong$ and $\prec$ can be seen as the "not equal to" and "less than" operators over two vectors. We can now define the equivalent concepts of local and global optimality in a MOP.

Definition 3: Local efficient (non-inferior) solution: A vector of objective values $F(\bar{x}),\ \bar{x} \in \theta(x)$ is said to be a local efficient solution of MOP **iff** $\nexists\ x \in (B_\delta(\bar{x}) \cap \theta(x))$ such that $F(x) \prec F(\bar{x})$ for some positive δ.

Definition 4: Global efficient (non-inferior) solution: A vector of objective values $F(\bar{x}),\ \bar{x} \in \theta(x)$ is said to be a local efficient solution of MOP **iff** $\nexists\ x \in \theta(x))$ such that $F(x) \prec F(\bar{x})$.

Definition 5: Pareto solutions: A point $\bar{x} \in \theta(x)$ is said to be a Pareto solution of MOP **iff** $F(\bar{x})$ is a global efficient solution of MOP.

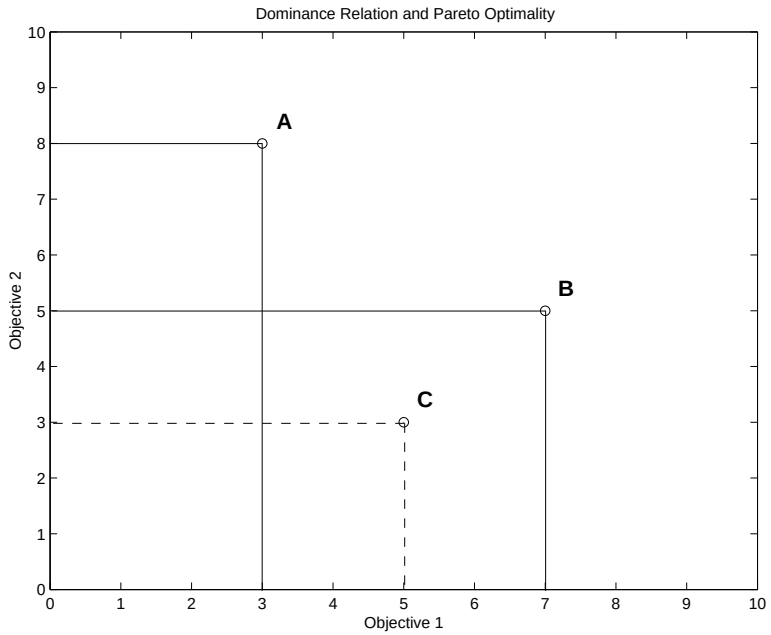


Figure 5.1: Diagram illustrating the concept of dominance and Pareto optimality. X-axis: Objective 1, Y-axis: Objective 2.

The concept of dominance and Pareto optimality is depicted in Figure 5.1. Let us consider the case where there are three solutions A, B, and C and assume that the two objectives 1 and 2 are to be maximized. A is not dominated by any other solution since it has the highest value for objective 2. Similarly, B is not dominated by any other solution since it has the highest value for objective 1. C is not dominated by A since it has a higher value for objective 1. However, C is

dominated by B since it has lower values for both objectives 1 and 2 compared to B. Hence we have the following situation

$$\begin{aligned} \mathsf{dominate}(A) &= \phi \\ \mathsf{dominate}(B) &= \phi \\ \mathsf{dominate}(C) &= \{B\} \end{aligned}$$

where $\mathsf{dominate}(C)$ denotes the set of solutions that dominate C. Therefore, the set of non-dominated or Pareto optimal solutions are given by

$$\mathsf{Pareto\ Set} = \{A, B\}$$

5.2 Evolutionary Multi-Objective Optimization

EMO combines the fields of evolutionary computation with multiple criteria decision-making for solving multi-objective optimization problems (Zitzler 1999; Deb 2001; Coello Coello, Van Veldhuizen, and Lamont 2002). EMO is an established sub-field of optimization and has been utilized for solving both theoretical and practical multi-objective optimization problems for over ten years (Zitzler 2002). A large range of practical applications of EMO to real-life problems across a host of different disciplines can be found in the reference texts by Deb (2001) and Coello Coello, Van Veldhuizen, and Lamont (2002). The literature surveyed on EMO covering general reviews, specific algorithms and related applications in the areas of robotics and artificial life is summarized in Table 5.1.

As explained in the preceding section, unlike in single-objective optimization, a multi-objective optimization problem gives rise to a number of optimal solutions, known as Pareto optimal solutions, of which none can be said to be better than the others with respect to all objectives. EAs are particularly suited for tackling multi-objective optimization problems by virtue of their population-based nature that allows for the generation of multiple solutions of the Pareto set within a single run (Deb 2001; Coello Coello, Van Veldhuizen, and Lamont 2002). Hence, the primary goal in EMO is to find or to approximate the set of Pareto optimal solutions through an evolutionary optimization process.

Type	Description	Reference
General	-	Goldberg (1989)
Reviews		Zitzler (1999)
		Van Veldhuizen and Lamont (2000a)
		Zitzler, Deb, and Thiele (2000)
		Deb (2001)
		Coello Coello, Van Veldhuizen, and Lamont (2002)
		Laumanns, Thiele, Deb, and Zitzler (2002)
		Zitzler (2002)
Algorithms	VEGA	Schaffer (1984)
	MOGA	Fonseca and Fleming (1993)
	NPGA	Horn, Nafpliotis, and Goldberg (1994)
	NSGA	Srinivas and Deb (1994)
	SPEA	Zitzler and Thiele (1999)
	ELSA	Menczer, Degeratu, and Street (2000)
	IEDS	Parmee, Cvetkovic, Watson, and Bonham (2000)
	MOMGA	Van Veldhuizen and Lamont (2000b)
	NSGA-II	Deb, Agrawal, Pratab, and Meyarivan (2000)
	PAES	Knowles and Corne (2000)
	MOMGA-II	Zydallis, Van Veldhuizen, and Lamont (2001)
	PDE	Abbass, Sarker, and Newton (2001); Abbass and Sarker (2002)
	MPANN	Abbass (2001; 2002a)
	SPEA2	Zitzler, Laumanns, and Thiele (2001)
	PCGA	Kumar and Rockett (2002)
	SPDE	Abbass (2002b)
Applications	Robotics & ICS	Gacogne (1997; 1999)
		Tan and Li (1997)
		Coello Coello, Christiansen, and Aguirre (1998)
		Dozier, McCullough, Homaifar, Tunstel, and Moore (1998)
		Pirjanian (1998; 2000)
		Tan, Lee, and Khor (1999)
		Leger (1999)
		Teo and Abbass (2002a)
		Teo, Nguyen, and Abbass (2003)
	A-Life	Oliveira, de Oliveira, and Omar (2000)
		Oliveira, Bortot, and de Oliveira (2002)
		Kim and Hallam (2002)
		Teo and Abbass (2002b; 2002c; 2003)

Table 5.1: Summary of literature survey on EMO reviews, algorithms and related applications in intelligent control systems (ICS), robotics and artificial life (A-Life).

The seminal work on EMO was that of Schaffer (1984) where the Vector Evaluation Genetic Algorithm (VEGA) was introduced for solving machine learning problems. Goldberg (1989) later outlined a 10-point list of how EMO algorithms can be formulated based on the concept of Pareto dominance, out of which a number of the early and well-known EMO algorithms were developed: Multi-Objective Genetic Algorithm (MOGA) (Fonseca and Fleming 1993), Non-dominated Sorting Genetic Algorithm (NSGA) (Srinivas and Deb 1994) and Niched Pareto Genetic Algorithm (NPGA) (Horn, Nafpliotis, and Goldberg 1994). These algorithms share two common properties in that solutions were ranked according to their dominance in the population and diversity was maintained using a niching strategy. However, these algorithms did not use any elite-preserving mechanism and as such, could not guarantee convergence to the Pareto optimal solutions. More recent algorithms have since focused on the use of elitism during the EMO process to improve on the convergence properties, such as Strength Pareto Evolutionary Algorithm (SPEA) (Zitzler and Thiele 1999), Pareto Archived Evolution Strategy (PAES) (Knowles and Corne 2000), Non-dominated Sorting Genetic Algorithm II (NSGA-II) (Deb, Agrawal, Pratab, and Meyarivan 2000), Multi-Objective Messy Genetic Algorithm (MOMGA) (Van Veldhuizen and Lamont 2000b), Strength Pareto Evolutionary Algorithm 2 (SPEA2) (Zitzler, Laumanns, and Thiele 2001) and Multi-Objective Messy Genetic Algorithm II (MOMGA-II) (Zydallis, Van Veldhuizen, and Lamont 2001).

A special issue of the Evolutionary Computation journal was published on EMO algorithms in 2000 (edited by Deb and Horn) where a number of seminal studies on EMO algorithms were presented. Firstly, multi-objective optimization problems were rigorously defined and the theoretical development of EMO algorithms was reviewed by Van Veldhuizen and Lamont (2000a). This article also presented an early attempt at classifying the different types of EMO algorithms and addressed specific issues such as fitness functions, Pareto ranking, niching, fitness sharing, mating restriction and secondary populations. A systematic comparison of a number of EMO algorithms was presented by Zitzler, Deb, and Thiele (2000)

using a set of six test functions specially chosen to elucidate particular problems associated with the EMO process. In this study, it was shown that elitism is particularly important for success in an EMO search. The PAES algorithm was introduced by Knowles and Corne (2000) as a simple (1 + 1) evolutionary strategy algorithm augmented with local search that is able to generate diverse solutions in solving multi-objective optimization problems. In this study, six variants of PAES were compared to variants of NPGA and NSGA over a diverse suite of six test functions. The results showed that PAES consistently performed well over the range of test functions. Parmee, Cvetkovic, Watson, and Bonham (2000) introduced the concept of an Interactive Evolutionary Design System (IEDS) as a methodology which allows EMO to be an interactive rather than a preset process. It was argued that such an interactive process permits the redefinition of the variable and objective space over the evolutionary process that will lead to a more finely tuned design environment. A simple selection method called local selection, which is based on the comparison of an individual's fitness to a fixed threshold rather than to another individual, was introduced by Menczer, Degeratu, and Street (2000) in an EMO algorithm called Evolutionary Local Selection Algorithm (ELSA). It was shown that ELSA naturally maintained genetic diversity by virtue of the local selection process and performed well for three multi-objective optimization problems.

More recently, the Pareto Converging Genetic Algorithm (PCGA) proposed by Kumar and Rockett (2002) eliminates the use of a niching strategy for diversity maintenance and was shown to produce competitive results on three benchmark problems while at the same time reducing computational cost. Laumanns, Thiele, Deb, and Zitzler (2002) also recently researched on the problem of maintaining diversity among the solutions while still being able to converge to the true Pareto optimal solutions in EMO algorithms. The concept of ϵ-dominance was proposed as a method for overcoming these problems and was shown that algorithms using this methodology for archiving solutions will theoretically converge to the actual Pareto-front in the limit while being able to maintain an optimal distribution of solutions along this front.

5.2.1 EMO in Control, Robotics and Artificial Life

EMO has been previously applied to the automated design of intelligent control systems by Tan and Li (1997) and Tan, Lee, and Khor (1999). There have also been studies on using *true* multi-objective optimization methods for the automatic design of artificial creatures. Pirjanian (1998, 2000) used multi-objective optimization to generate action selection modules in a behavior-based robotics experiment. However, this study utilized conventional mathematical optimization methods and did not make use of an evolutionary optimization approach. Evolutionary methods have been used to solve navigational problems with multiple objectives for 2D mobile agents in simulation (Dozier, McCullough, Homaifar, Tunstel, and Moore 1998; Gacogne 1997; Gacogne 1999). Coello Coello, Christiansen, and Aguirre (1998) also used an EMO approach for a robotics design problem but this experiment involved only a non-autonomous subject in the form of an attached robotic manipulator arm. Multi-objective evolutionary optimization has also been used by Leger (1999) although the focus of the EMO approach was for optimizing the physical configurations of modular robotic components rather than for the generation of autonomous robotic controllers. There have been a number of other studies involving the use of some form of EMO for the design of robotic manipulator arms as reviewed by Coello Coello, Van Veldhuizen, and Lamont (2002).

More recently, Oliveira, Bortot, and de Oliveira (2002) used an EMO approach in an artificial life study of 1D cellular automata for the density classification task problem. It was reported that the use of an EMO approach offered significant advantages in terms of defining and evaluating the fitness of evolving populations over a weighted sum approach carried out in a prior experiment (Oliveira, de Oliveira, and Omar 2000). Kim and Hallam (2002) also recently reported the use of EMO for solving the so-called *Woods* problem, which are goal-search problems for agents starting at random initial positions and having to find an end-state goal position. The objective of the study was to quantify the amount of internal memory states required for the finite state machine controllers to solve a given *Woods* task. Pareto-fronts of discrete internal memory states were minimized in a

trade-off against maximizing the fitness of the agents, which were evaluated as the minimum number of steps required to reach the end-state goal position from the initial starting position. However, the artificial creatures were only very simple 2D agents that acted in a discrete grid-world environment with movement allowed only in the four cardinal directions. In this chapter, we will demonstrate the use of EMO for evolving completely autonomous, embodied and situated creatures that act in a 3D world with fully continuous and non-restrictive movements that trade-off between the number of internal nodes required in the neural network controller and locomotion capability achieved. Furthermore, our experiments are aimed at generating legged locomotion in 3 dimensions rather than wheeled or mobile locomotion behaviors that are restricted to 2 dimensions.

5.3 PDE Algorithm

Abbass et al. first introduced the Pareto-frontier Differential Evolution (PDE) algorithm for vector optimization problems (Abbass and Sarker 2002; Abbass, Sarker, and Newton 2001). PDE is a multi-objective adaptation of the original *Differential Evolution* (DE) algorithm introduced by Storn and Price (1995) for optimization problems over continuous domains. The PDE algorithm outperformed the SPEA algorithm (Zitzler and Thiele 1999) on five benchmark problems in this introductory investigation.

PDE combined with local search was later introduced for evolving ANNs in the MPANN algorithm (Abbass 2001). MPANN was found to be highly effective for knowledge discovery in databases. In subsequent work, the MPANN algorithm was empirically shown to possess better generalization in medical diagnosis of breast cancer whilst incurring a much lower computational cost (Abbass 2002a).

In an extension to PDE, a self-adaptive version called Self-adaptive Pareto Differential Evolution (SPDE) algorithm was proposed to allow for self-adaptation of mutation and crossover rates during the optimization process (Abbass 2002b). Both rates for new individuals are inherited from parents during crossover and mutated

in the same process that occurs for decision variables. SPDE was found to be highly competitive against 13 other EMO algorithms on four benchmark test functions and actually outperformed a number of current state-of-the-art algorithms.

As described above, the SPDE algorithm has been found to be a highly effective algorithm for optimization over a continuous domain. Moreover, it has also been tested successfully for the evolution of ANNs. For these reasons, SPDE was chosen as the algorithm for evolving the creature's controllers since the parameters that are being optimized in the evolutionary process are the real-valued weights of the neural network. Furthermore, it provides an added advantage over the original PDE algorithm since it allows for self-adaptation of the crossover and mutation rates. It should be noted that other EMO algorithms may also be used to evolve the creature's controllers. However, since the objective of this work is to investigate the application of an EMO approach for evolving artificial creature controllers and not a comparison between EMO algorithms, the question of which EMO algorithm will work best for this purpose is beyond the scope of this thesis and remains an open question for future work.

5.4 Proposed SPDE-Based Controller Evolution

More recently, the SPDE algorithm has been combined with MPANN for evolving artificial neural networks called the Self-adaptive Pareto Artificial Neural Network (SPANN) algorithm (Abbass 2003). In this thesis, we propose a modified version of SPANN for controller evolution. There are two major differences between this proposed version and the original version of SPANN. Firstly, SPANN uses back-propagation for learning. In the case of locomotion controller evolution, the task to be learned is not clear-cut as in canonical classification problems. As such, the only learning that takes place in the modified version of SPANN occurs only through evolutionary adaptation. A possible future work would be to investigate the possibility of augmenting the current proposed version of SPANN with lifetime learning by somehow introducing a measure of error associated with the locomotion task,

which would then allow back-propagation to be used. Nolfi (1999) has previously shown how learning through back-propagation can be integrated with evolutionary artificial neural networks to predict the next move state for an autonomous agent in a 2D grid world. However, the prediction task required in this case involved only movements in the four cardinal directions by virtue of the grid world. As such it remains an open question how such a methodology can be applied to 3D physically accurate embodied creatures living in virtual worlds with infinitely large numbers of possible directions for locomotion.

Secondly, the repair function originally used in SPANN for evolving the crossover and mutation rates (which truncates the whole number portion leaving only the decimal portion), though useful for the data mining task, was found to cause premature convergence of these rates to the lower boundary of 0 when evolving controllers. Consequently, the evolutionary optimization process would also prematurely stagnate due to the lack of crossover and mutation during reproduction. Hence a new repair function as explained in Section 5.4.1 is proposed in the modified version of SPANN to overcome this problem. For the remainder of the thesis, this proposed version of the SPANN algorithm for controller evolution is referred to whenever the acronym SPANN is used. The pseudocode of this proposed version of the algorithm is given in the next subsection.

5.4.1 The SPANN Algorithm

The pseudocode for the SPANN algorithm is as follows:

1. Create a random initial population of potential solutions. The elements of the weight matrix Ω are assigned random values according to a Gaussian distribution $N(0, 1)$. The elements of the binary vector ρ are assigned the value 1 with probability 0.5 based on a randomly generated number according to a uniform distribution between $[0, 1]$, otherwise 0. The crossover rate δ and mutation rate η are assigned random values according to a uniform distribution between $[0, 1]$.

2. Repeat

   (a) Evaluate the individuals in the population and label those who are non-dominated.

   (b) If the number of non-dominated individuals is less than 3 repeat the following until the number of non-dominated individuals is greater than or equal to 3:

      i. Find a non-dominated solution among those who are not labelled.

      ii. Label the solution as non-dominated.

   (c) Delete all dominated solutions from the population.

   (d) Repeat

      i. Select at random an individual as the main parent α_1, and two individuals, α_2, α_3 as supporting parents.

      ii. Select at random a variable j.

      iii. **Crossover:** With some probability $Uniform(0,1) > \delta^{\alpha_1}$ or if $i = j$, do

$$\omega_{ih}^{child} \leftarrow \omega_{ih}^{\alpha_1} + N(0,1)(\omega_{ih}^{\alpha_2} - \omega_{ih}^{\alpha_3}) \tag{5.1}$$

$$\omega_{ho}^{child} \leftarrow \omega_{ho}^{\alpha_1} + N(0,1)(\omega_{ho}^{\alpha_2} - \omega_{ho}^{\alpha_3}) \tag{5.2}$$

$$\rho_h^{child} \leftarrow \begin{cases} 1 & if\ (\rho_h^{\alpha_1} + N(0,1)(\rho_h^{\alpha_2} - \rho_h^{\alpha_3})) \geq 0.5 \\ 0 & otherwise \end{cases} \tag{5.3}$$

$$\delta^{child} \leftarrow \delta^{\alpha_1} + N(0,1)(\delta^{\alpha_2} - \delta^{\alpha_3}) \tag{5.4}$$

$$\eta^{child} \leftarrow \eta^{\alpha_1} + N(0,1)(\eta^{\alpha_2} - \eta^{\alpha_3}) \tag{5.5}$$

      otherwise

$$\omega_{ih}^{child} \leftarrow \omega_{ih}^{\alpha_1} \tag{5.6}$$

$$\omega_{ho}^{child} \leftarrow \omega_{ho}^{\alpha_1} \tag{5.7}$$

$$\rho_{h}^{child} \leftarrow \rho_{h}^{\alpha_1} \tag{5.8}$$

$$\delta^{child} \leftarrow \delta^{\alpha_1} \tag{5.9}$$

$$\eta^{child} \leftarrow \eta^{\alpha_1} \tag{5.10}$$

where each variable in the main parent is perturbed by adding to it a Gaussian value $N(0,1)$ multiplied by the difference between the two values of this variable in the two supporting parents. At least one variable in Ω must be changed. If δ or η are not in $[0,1]$, repair by adding (if < 0) or subtracting (if > 1) a random number between $[0,1]$ until δ and η are in $[0,1]$.

iv. **Mutation:** With some probability $Uniform(0,1) > \eta^{\alpha_1}$, do

$$\omega_{ih}^{child} \leftarrow \omega_{ih}^{child} + N(0, \eta^{\alpha_1}) \tag{5.11}$$

$$\omega_{ho}^{child} \leftarrow \omega_{ho}^{child} + N(0, \eta^{\alpha_1}) \tag{5.12}$$

$$\rho_{h}^{child} \leftarrow \begin{cases} 1 & if\ \rho_{h}^{child} = 0 \\ 0 & otherwise \end{cases} \tag{5.13}$$

$$\delta^{child} \leftarrow N(0,1) \tag{5.14}$$

$$\eta^{child} \leftarrow N(0,1) \tag{5.15}$$

(e) Until the population size is M

3. Until maximum number of generations is reached.

5.5 Experimental Setup

Four series of experiments were conducted to compare the evolution of controllers using the four different types of ANN architecture. The fitness of each genotype in these experiments was evaluated according to both the f_1 and f_2 objective functions, which measures the locomotion distance achieved and number of hidden units used by the controller respectively as defined in Section 3.4.1. The evolutionary and simulation parameters used were as reported in Section 3.5: 1000 generations, 30 individuals, 500 timesteps and 10 repeated runs. As with the fitness landscape experiments in Chapter 4, the maximum number of hidden units allowed in the ANN was set to 15. Being the objects of the evolutionary optimization process, the locomotion distance and number of hidden units used in the ANN were recorded for every individual generated in every generation.

First, we analyze the optimization results from the evolution of creature controllers for the four types of ANN architectures. Next, we compare the controllers obtained using our SPANN algorithm against those obtained from using the random search, hill-climbing and random walk algorithms. Then, we analyze the evolutionary dynamics at the individual as well as population level of genotypes generated during the evolutionary optimization process to provide a deeper insight into how the evolution of controllers affects the evolution of locomotion capabilities in a physically simulated artificial creature. This is followed by a characterization of the search space difficulty associated with each of the four types of ANN architectures to investigate whether any of the four ANN architectures provide any significant advantages in terms of evolutionary search for controllers with high locomotion fitness. Finally, we analyze the operational dynamics of the overall best controller evolved for locomotion distance using the SPANN algorithm to ascertain what is actually happening in the creature's limbs as they are controlled by the evolved ANN during locomotion as well as the effect of noise on the performance of the evolved ANN controller.

5.6 Results and Discussion

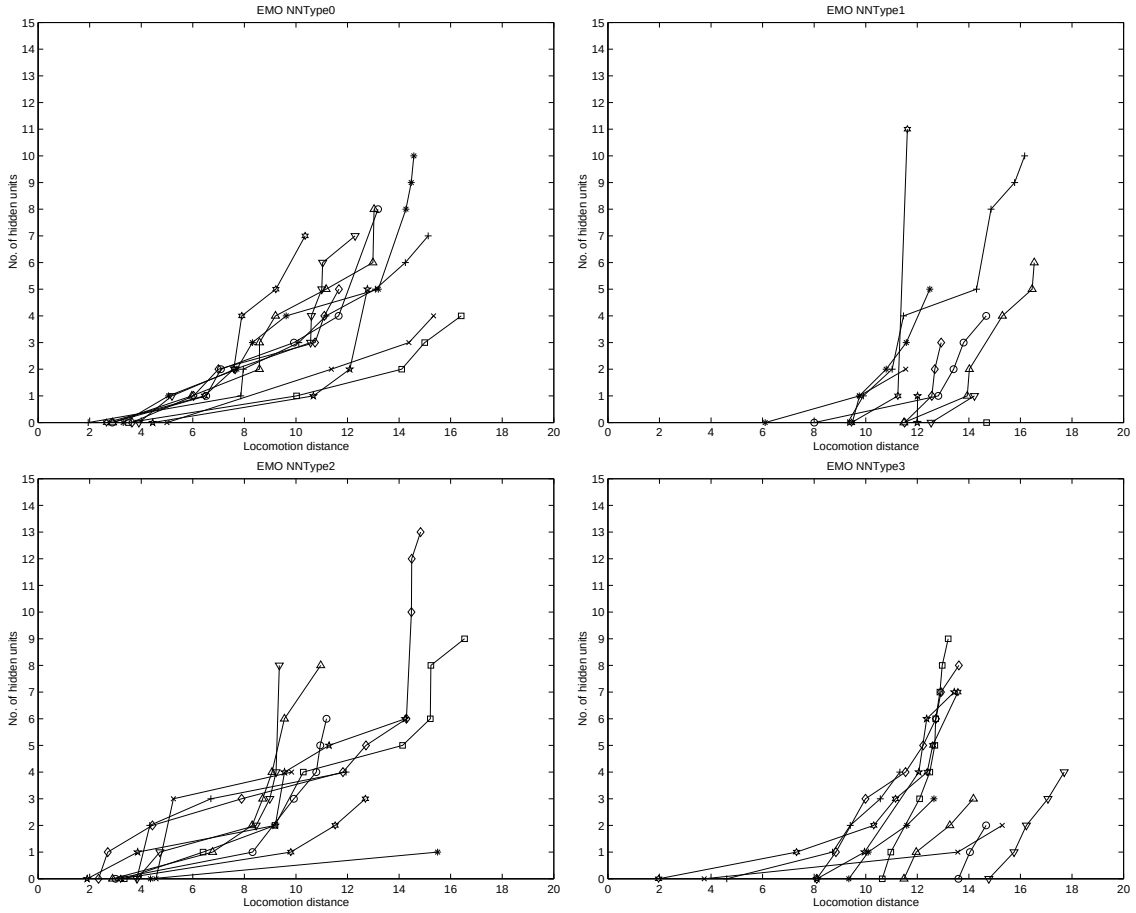


Figure 5.2: Pareto-front of solutions obtained for 10 runs using the SPANN algorithm for ANN architecture 1. NNType0 (top left), 2. NNType1 (top right), 3. NNType2 (bottom left), 4. NNType3 (bottom right). X-axis: Locomotion distance, Y-axis: No. of hidden units.

The Pareto-fronts achieved at the last generation for each of the 10 runs are plotted in Figure 5.2. It can be noticed that a variety of solutions in terms of controller size and locomotion capability was obtained in the majority of the evolutionary runs. Most of the solutions on the Pareto-frontier comprised of controllers with less than 5 hidden units in the ANN. This is an indication that larger networks did not offer significant advantages in terms of generating better locomotion capabilities compared to smaller networks. There were no obvious differences between

the four different types of ANN architectures. However, having direct input-output connections did have an observable effect on networks with 0 hidden units. The locomotion achieved with NNType1 (Figure 5.2.2) and NNType3 (Figure 5.2.4) architectures ranged between 2 up to almost 15 whereas NNType0 (Figure 5.2.1) and NNType2 (Figure 5.2.3) architectures were clustered between 2 to 4. It should be pointed out that although NNType0 and NNType2 networks with 0 hidden units do not have any sensor-to-motor mappings whatsoever, some small movement is still achieved due to the initial forces generated from the outputs of the these networks since an activation value of zero would still produce an output signal of 0.5 by virtue of the sigmoidal transfer function — however, without any mapping between the input and output layers of the networks, this initial movement is unsustainable due to the lack of synchronization ability (Teo and Abbass 2002a). Hence, NNType1 and NNType3 controllers with direct input-output connections could achieve sufficiently good locomotion capabilities without requiring a hidden layer. The ability of such *pure reactive* agents for solving complex sensory-motor coordination tasks have previously been reported in wheeled robots (Lund and Hallam 1997; Pasemann, Steinmetz, Hulse, and Lara 2001a; Nolfi 2002). These direct connections between the input and output layers also appeared to have generated Pareto optimal networks with smaller sizes in a large majority of the runs. This may be due to the fact that the direct input-output connections are already providing a good mechanism for basic locomotion, thus requiring only a few extra hidden units to further improve on this basic locomotion.

The largest network on the Pareto-front can be found with the NNType2 architecture using up to 13 hidden units. This suggests that the recurrency is somehow making the locomotion task more difficult to learn and thereby adding unnecessary complexity to the task (it must be remembered that the recurrent connections in NNType2 are already themselves adding structural complexity to the ANN architecture). This observation is supported by comparing against the Pareto optimal controllers obtained using the simple feed-forward-only NNType0 architecture, a number of which utilized fewer hidden units than the NNType2 Pareto optimal

controllers but were still able to achieve highly similar locomotion capabilities.

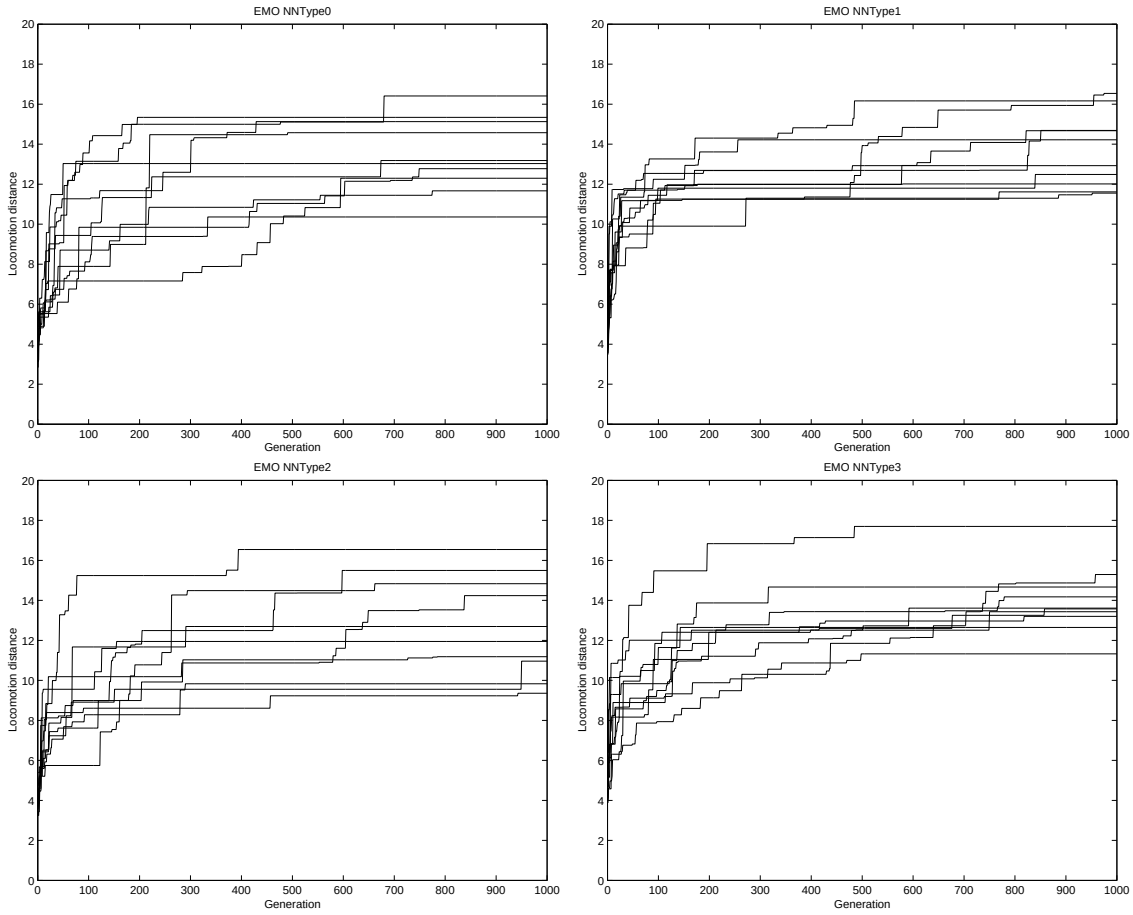


Figure 5.3: Best locomotion distance of Pareto solutions obtained over 1000 generations for 10 runs using the SPANN algorithm for ANN architecture 1. NNType0 (top left), 2. NNType1 (top right), 3. NNType2 (bottom left), 4. NNType3 (bottom right). X-axis: Generation, Y-axis: Locomotion distance.

The evolution of the Pareto solution for best locomotion distance using the SPANN algorithm for the 10 runs over 1000 generations is depicted in Figure 5.3. No marked differences between the four ANN architectures could be seen. Most of the improvement occurred early during the evolutionary process where in most runs, the best solution exceeded a locomotion capability of beyond 10 units by the 150th generation. Compared to the best solutions obtained using random search, hill-climbing and random walk, the EMO algorithm provided a smoother progress

over time in discovering fitter solutions. This may be an indication that the landscape is explored better using EMO. Among the four different ANN architectures, NNType1 appeared to have the least amount of variation in terms of the best solution obtained over the 10 different runs (Figure 5.3.2). On the other hand, NNType2 showed much larger differences between the 10 runs (Figure 5.3.3). This might be an indication that the additional direct input-output connections exhibited a less rugged landscape compared to additional recurrent-only connections in the ANN when evolving controllers using the EMO algorithm. In the former case, most of the runs were able to proceed along more similar paths due to the presence of a smoother landscape, thereby leading to less variation among solutions.

NNType	Overall Best Locomotion Distance	Average Best Locomotion Distance ± Standard Deviation	t-statistic (against NNType0)
0	16.4104	13.4755 ± 1.8613	-
1	16.5375	13.6866 ± 1.8318	0.30
2	16.5418	12.7079 ± 2.4674	(0.87)
3	17.6994	13.9626 ± 1.7033	0.53

Table 5.2: Comparison of best locomotion distance for Pareto solutions found using the SPANN algorithm over 10 independent runs.

The overall best Pareto solution in terms of locomotion fitness together with the mean best locomotion fitness and standard deviations obtained using the SPANN algorithm are given in Table 5.2. The overall and average best solutions were obtained with the NNType3 architecture although the differences between architectures were small. A t-test at both $\alpha = 0.05$ and $\alpha = 0.01$ significance levels showed no significant differences between the four ANN architectures in terms of the best solutions obtained over 10 independent runs. The lowest average best locomotion fitness was given by NNType2, which also showed a much larger deviation among the best solutions found compared to the other architectures. This supports the earlier observation that the evolutionary path encountered when using additional recurrent-only connections in NNType2 was more rugged, leading to greater variation among solutions.

Table 5.3 lists the global Pareto optimal solutions found by SPANN over

NNType	No. of Hidden Units	Locomotion Distance
0	0	4.9981
	1	10.6792
	2	14.1010
	3	14.9951
	4	16.4104
1	0	14.6870
	4	15.2998
	5	16.4548
	6	16.5375
2	0	4.5740
	1	15.4944
	9	16.5418
3	0	14.7730
	1	15.7506
	2	16.2295
	3	17.0663
	4	17.6994

Table 5.3: Comparison of number of hidden units used and locomotion distance for global Pareto optimal controllers obtained using the SPANN algorithm over 10 independent runs.

the 10 runs for each type of ANN architecture. The solutions with the highest locomotion fitness used between 4 and 9 hidden units in the ANN controller. For NNType0 and NNType3, the maximum number of hidden units required to generate the best locomotion was only 4 hidden units whereas NNType1 required 6 hidden units. NNType2 required the most number of hidden units (9). It is interesting to note that by allowing direct input-output connections in NNType1 and NNType3, controllers which did not use the hidden layer at all (0 hidden units) could generate a sufficiently good locomotion ability, moving the creature up to a distance of 14.7 and 14.8 units respectively. This is empirical proof that perceptron-like controllers, which rely only on input-to-output connections without any internal nodes, are sufficient for generating simple locomotion in a four-legged artificial creature. As previously pointed out, this phenomenon has been previously observed to occur in wheeled robots as well (Lund and Hallam 1997; Pasemann, Steinmetz, Hulse, and Lara 2001a; Nolfi 2002). As such, robots that are only required to perform simple

tasks can be evolved as purely reactive agents, which would dramatically simplify the process of synthesizing successful robot controllers. As previously explained in the first paragraph of this section, for NNType0 and NNType2 controllers with 0 hidden units, there is still an act of force on the creature produced by the zero-activation output from these networks that permit the small initial movements.

5.6.1 SPANN vs. Random Search, Hill-Climbing and Random Walk

NNType	Algorithm	Average Best Locomotion Distance ± Standard Deviation	t-statistic (against SPANN)	No. of Hidden Units
0	SPANN	13.4755 ± 1.8613	-	6.5 ± 2.0
	Random Search	7.5406 ± 0.6733	(9.96)	10.0 ± 2.3
	Hill-Climbing	5.5969 ± 0.9714	(11.39)	7.0 ± 2.3
	Random Walk	9.7725 ± 1.2009	(4.63)	7.4 ± 2.1
1	SPANN	13.6866 ± 1.8318	-	4.1 ± 3.5
	Random Search	10.0931 ± 0.7348	(6.06)	7.4 ± 4.0
	Hill-Climbing	8.4652 ± 2.6246	(5.45)	7.4 ± 2.0
	Random Walk	10.4776 ± 0.9616	(4.84)	7.4 ± 2.2
2	SPANN	12.7079 ± 2.4674	-	6.2 ± 3.5
	Random Search	7.3609 ± 0.7490	(7.02)	10.9 ± 2.5
	Hill-Climbing	7.6568 ± 2.9365	(3.49)	6.4 ± 2.5
	Random Walk	9.5602 ± 1.3372	(3.59)	8.2 ± 2.0
3	SPANN	13.9626 ± 1.7033	-	4.9 ± 2.6
	Random Search	10.2878 ± 1.2747	(5.59)	9.2 ± 4.2
	Hill-Climbing	6.9057 ± 1.5719	(11.46)	6.8 ± 2.2
	Random Walk	10.0333 ± 1.2535	(5.78)	7.0 ± 1.4

Table 5.4: Comparison of best locomotion distance for Pareto/best solutions obtained over 10 independent runs using the SPANN, random search, hill-climbing and random walk algorithms over 10 independent runs.

Table 5.4 provides a comparison of the best results for locomotion distance obtained using the SPANN, random search, hill-climbing and random walk algorithms. For all four ANN architectures, the SPANN algorithm produced controllers that had much higher locomotion capabilities than controllers obtained using random search, hill-climbing and random walk. To confirm that the results obtained

using the proposed SPANN algorithm were significantly superior, a t-test was conducted. At both significance levels of $\alpha = 0.05$ and $\alpha = 0.01$, the best solutions obtained using SPANN over 10 independent runs were all significantly higher in terms of locomotion fitness than random search, hill-climbing and random walk across all four ANN architectures.

The last column of Table 5.4 shows the average number of hidden units used in the ANN of the best controller evolved for locomotion distance obtained over the 10 different runs. It is clear that with the additional optimization objective of minimizing the number of hidden units used in the ANN for the SPANN algorithm, the number of hidden units required by the best controllers obtained from evolution for all four different types of ANN architectures were lower than those obtained using random search, hill-climbing and random walk, which were just optimizing along the single objective of locomotion distance. However, what is interesting is the fact that a significantly higher locomotion distance was achieved with a smaller controller size when the EMO algorithm was used. This may be explained by the strong evolutionary pressures on the survival of controllers within the population during reproduction and selection, thereby playing a significant role in forcing controllers to become increasingly efficient at locomotion as well as requiring fewer active hidden units in the ANN controller. Moreover, the inclusion of a second objective to the optimization process may have provided an extra-dimensional bypass in which the SPANN algorithm was able to reach a fitter solution space compared to random search, hill-climbing and random walk. This phenomenon has been previously encountered during the evolution of walking behavior in a simulated biped robot when additional morphological parameters for size and mass distribution of body segments were added to the original chromosome of the artificial creature (Bongard and Paul 2001). The EMO methodology may have naturally created the extra-dimensional bypass through the optimization of multiple objectives. The extra-dimensional bypass may have also arisen from the usage of ρ, which allows hidden units to continuously evolve even though it may be inactive during certain periods of the evolutionary process.

5.6.2 Evolutionary Dynamics

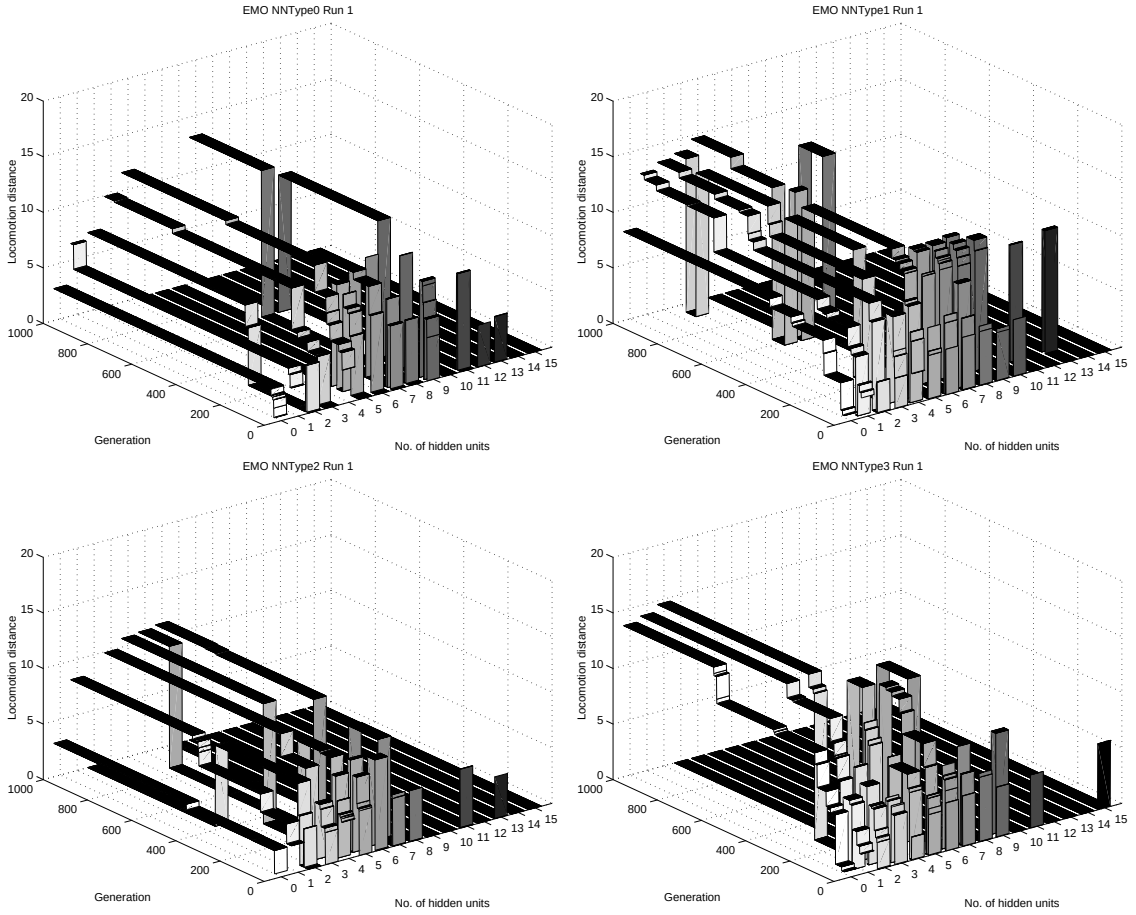


Figure 5.4: Non-dominated solutions generated by the SPANN algorithm over 1000 generations for runs using the first seed for ANN architecture 1. NNType0 (top left), 2. NNType1 (top right), 3. NNType2 (bottom left), 4. NNType3 (bottom right). X-axis: Generation, Y-Axis: No. of hidden units, Z-axis: Locomotion distance. Additional graphs can be found in the accompanying CD-ROM.

Figure 5.4 illustrates the evolution of non-dominated solutions over 1000 generations from the first run using the four different ANN architectures. Three main results can be concluded from the analysis of these graphs, which are representative of the dynamics of controller evolution from the other 9 runs conducted for each of the architecture types respectively. Firstly, a large variety of solutions in terms of locomotion capability and controller size is maintained throughout the evolutionary

process. The evolution starts off with an even spread of solutions across the range of permissible hidden layer sizes. A high level of evolutionary activity can then be seen to occur before the 250–300th generation. During this evolutionary era, solutions with different controller sizes and locomotion capabilities were actively competing for survival as non-dominated solutions that will be carried over to the next generation as parents. After the initial flurry of evolutionary activity, the optimization process begins to stabilize after the 400–500th generation. Even at the end of 1000 generations, between 3 to 6 different Pareto controllers in terms of the number of hidden units used in the ANN were still present in the optimal set of solutions out of a possible 16 different configurations for active hidden units.

Secondly, the graphs show that some genotypes with a certain hidden layer architecture disappears from the non-dominated set of solutions and then reappears, for example controllers with 8 hidden units in NNType0 (Figure 5.4.1) and controllers with 5 hidden units in NNType2 (Figure 5.4.3). This phenomenon is indicative of the evolutionary search process moving through the fitness landscape by experimenting with different hidden layers sizes and eventually re-discovering a network configuration previously used but now with added locomotion capabilities.

Lastly, we see from the evolutionary dynamics of controller evolution that it is generally very hard for larger controllers with more hidden units to survive due to the strong evolutionary pressure of trading-off the ANN performance and its complexity. This observation is attributed to the fact that a larger controller does not easily lead to locomotion abilities that can't be achieved with a smaller controller in this particular problem. As a result, larger controllers find it hard to compete with smaller controllers in trying to maximize the horizontal distance travelled by the quadruped.

In summary, the multi-objective approach maintains genetic diversity by allowing individuals with different controllers and capabilities to be retained within the genetic pool. This shows that the EMO approach is highly advantageous since it allows for simultaneous maintenance of genetic diversity as well as survival of individuals that exhibit particular advantages over the rest of the population. The

maintenance of genetic diversity using an EMO approach has recently been verified by Abbass and Deb (2003).

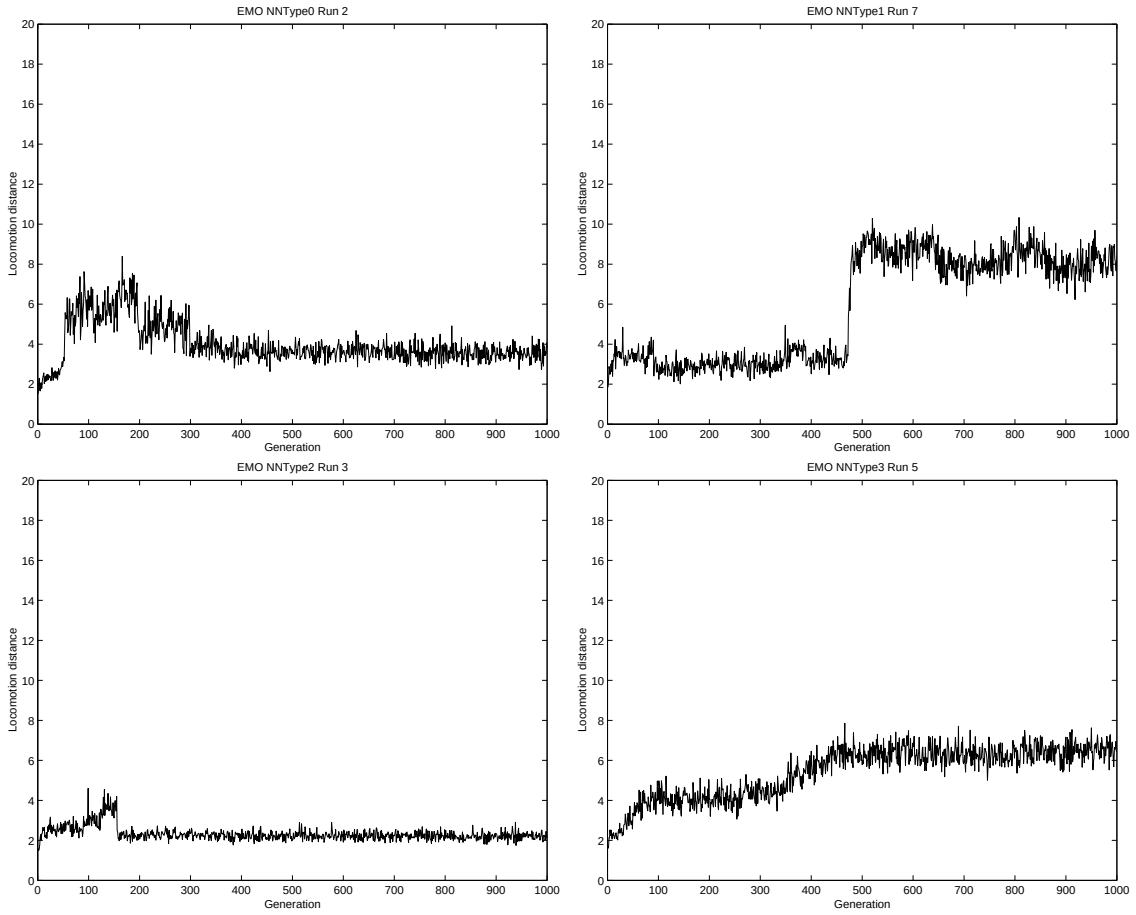


Figure 5.5: Mean locomotion distance of population over 1000 generations (selected seeds only) using the SPANN algorithm for ANN architecture 1. NNType0 (top left), 2. NNType1 (top right), 3. NNType2 (bottom left), 4. NNType3 (bottom right). X-axis: Generation, Y-Axis: Locomotion distance. Additional graphs can be found in the accompanying CD-ROM.

Next, we analyze the mean of the population for locomotion distance as it evolved over 1000 generations, which is depicted in Figure 5.5. There were generally four trends in the movement of the population mean during the evolutionary optimization process present across the four types of ANN architectures. The most commonly occurring trends in the population means were that of an early increase

followed by a decrease and then stabilizing into a constant range, which is depicted in Figure 5.5.1, and that of a slow and small increase over time, which is depicted in Figure 5.5.4. However, there were also instances where there was a sudden evolutionary jump into a much higher fitness space which was then maintained throughout the evolutionary process, as shown in Figure 5.5.2. This can be explained by reaching an evolutionary peak and maintaining the search process within this high fitness subspace, perhaps assisted by the presence of a large basin of attraction associated with this subspace. This occurrence may also be indicative that the majority of new solutions being generated are located near or close to a newly-found superior solution, which results in the increase in population mean since new solutions being generated have fitness values close to this superior individual. Finally, there were also occurrences of the opposite phenomenon, that of a sudden decrease in the population mean, indicative of reaching an evolutionary peak which is surrounded by low fitness subspaces, as illustrated in Figure 5.5.3. This again suggests the presence of a large basin of attraction, this time associated with a low fitness subspace. Overall, these results support the earlier characterization of the random walk fitness landscape using informational measures (Section 4.5.3.1) which showed that there were a variety of shapes present on the evolutionary landscape and that depending on the initialization and trajectory of the search process, smooth or highly rugged landscape regions may be encountered.

NNType	Average Locomotion Distance ± Standard Deviation
0	3.9848 ± 0.7972
1	5.5943 ± 1.9875
2	3.4466 ± 1.2297
3	5.1650 ± 1.7368

Table 5.5: Mean and standard deviation of all individuals' f_1 fitness generated within all generations and over all 10 runs for SPANN. The average is calculated over 300,000 individual f_1 fitness recorded across the entire evolutionary optimization process.

Table 5.5 lists the average locomotion fitness achieved by all individuals generated throughout the 1000 generations over 10 runs. On average, the fitness

space occupied by the evolving population was highest in NNType1 followed closely by NNType3. Correspondingly, a much lower fitness region of the search space was occupied by the evolving populations of NNType0 and NNType2. This indicates that the NNType1 and NNType3 architectures, which have the direct input-output connections, permitted reproduction of offspring that were generally fitter than those produced by NNType0 and NNType2. The inclusion of direct input-output connections introduces some sort of a lower bound on the performance of controllers having the NNType1 and NNType3 architectures since these direct input-output connections are not subjected to evolutionary pressures and hence are always present in any controller of these types. On the other hand, controllers using NNType0 and NNType2 architectures have no such lower bound since controllers of these types that use no hidden units will not have any mapping between the input and output layers whatsoever. Nonetheless, it should be noted that although the fitness regions occupied by the evolving populations of NNType1 and NNType3 were higher than that of NNType0 and NNType2, no significant advantages were evident in terms of leading the search towards a more optimal final solution since a t-test showed that there were no significant differences between the best locomotion distances of the Pareto solutions found using the four different types of ANN architecture (see Table 5.2).

5.6.3 Search Space Characterization

The distribution of genotypes generated during the EMO search process is plotted in Figure 5.6 in terms of locomotion distance and number of hidden units used in the ANN. Except for NNType1, the distribution of solutions obtained across the solution space was less clustered for all other architectures using the EMO algorithm compared to random search, hill-climbing and random walk. This suggests that the EMO search process was able to sample more uniformly across both objective spaces in spite of having the added optimization criterion of minimizing the number of hidden units used in the ANN. In NNType1, a number of spikes in the frequency distribution could be seen (Figure 5.6.2). A separate graph is

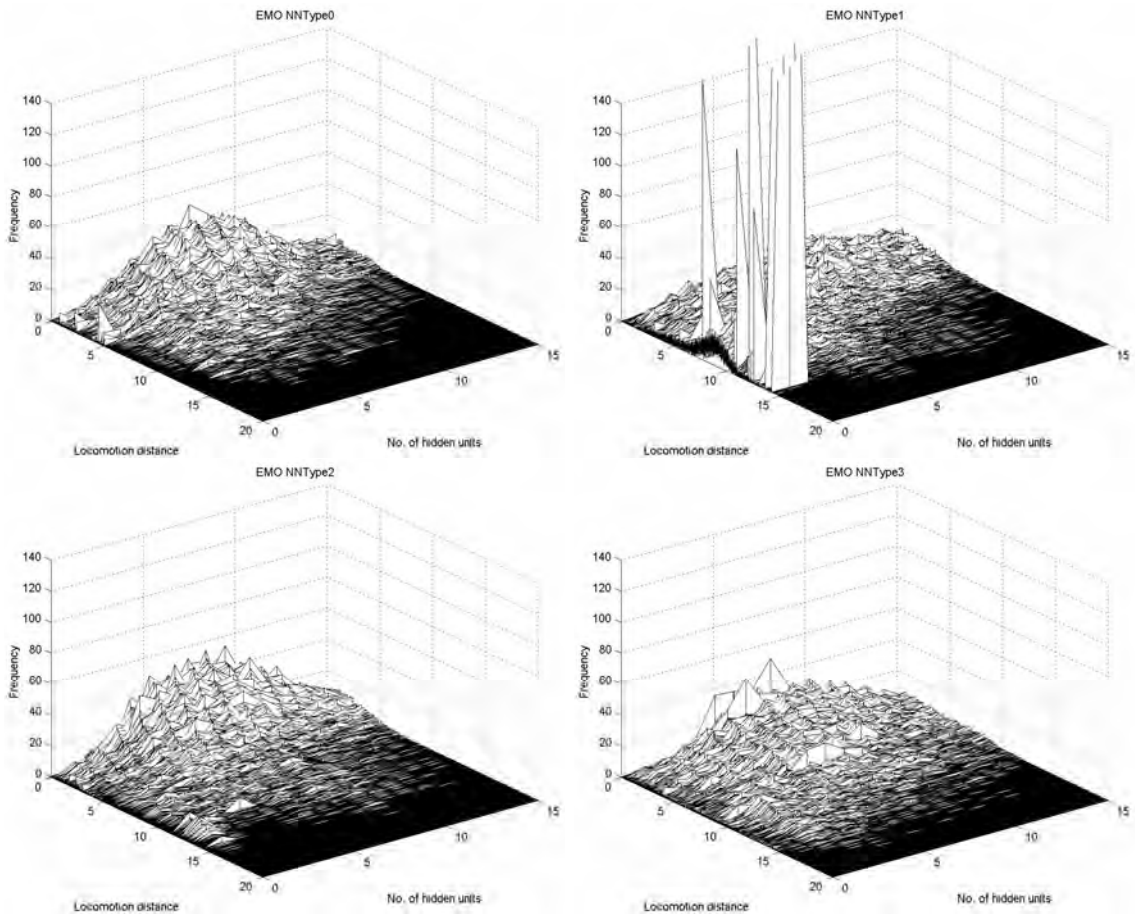


Figure 5.6: Frequency distribution of solutions using the SPANN algorithm for ANN architecture 1. NNType0 (top left), 2. NNType1 (top right), 3. NNType2 (bottom left), 4. NNType3 (bottom right). X-axis: Locomotion distance, Y-axis: No. of hidden units, Z-axis: Frequency.

plotted for this architecture in Figure 5.7. This occurrence is discussed further in the next paragraph where the concentration of solutions in terms of the two separate objectives is more evident in the accompanying contour graphs. Furthermore, for all four architectures, a significantly larger proportion of individuals were generated in the fitter regions of the locomotion objective space compared to random search, hill-climbing and random walk.

The contour graphs in Figure 5.8 illustrate the distribution of solutions across the two objectives of minimizing the hidden layer and maximizing locomotion

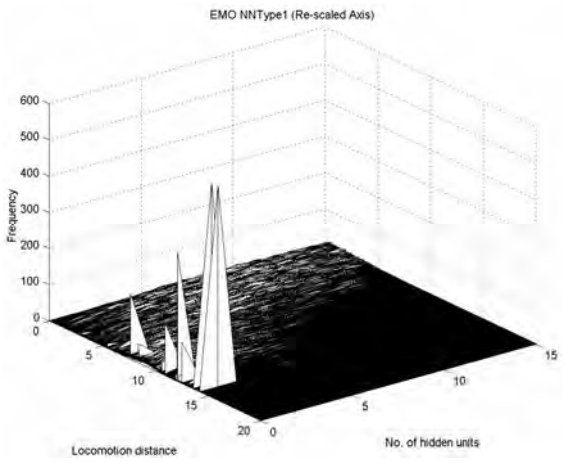


Figure 5.7: Frequency distribution of solutions obtained using the SPANN algorithm for ANN architecture NNType1 with re-scaled frequency axis. X-axis: Locomotion distance, Y-axis: No. of hidden units, Z-axis: Frequency.

distance. Compared to hill-climbing and random walk, the effect of having an extra objective in the form of minimizing the number of hidden units used in the ANN can be seen from the slight shift in distribution of solutions towards the lower halves of the graphs, which is very prominent in NNType3 (Figure 5.8.4) and most obvious in NNType1 (Figure 5.8.2). As was discussed previously, the presence of direct input-output connections allowed the search process to find effective locomotion controllers using a very small number of hidden units or none at all. For NNType1, the spikes highlighted in the previous paragraph can be seen in the high fitness regions of both objective spaces using only 0 or 1 hidden units and achieving between 9 and 15 units of locomotion distance.

The probability density function of solutions obtained using the SPANN algorithm is illustrated in Figure 5.9 for all four ANN architectures. The graphs clearly show that the probability of encountering fitter solutions in terms of locomotion distance was much higher than in all previous search algorithms. The probability density curves were similar to a large extent across all four architectures in terms of the locomotion fitness. From the cumulative curve, it can be seen that for NNType0 (Figure 5.9.1) and NNType2 (Figure 5.9.3), the probability

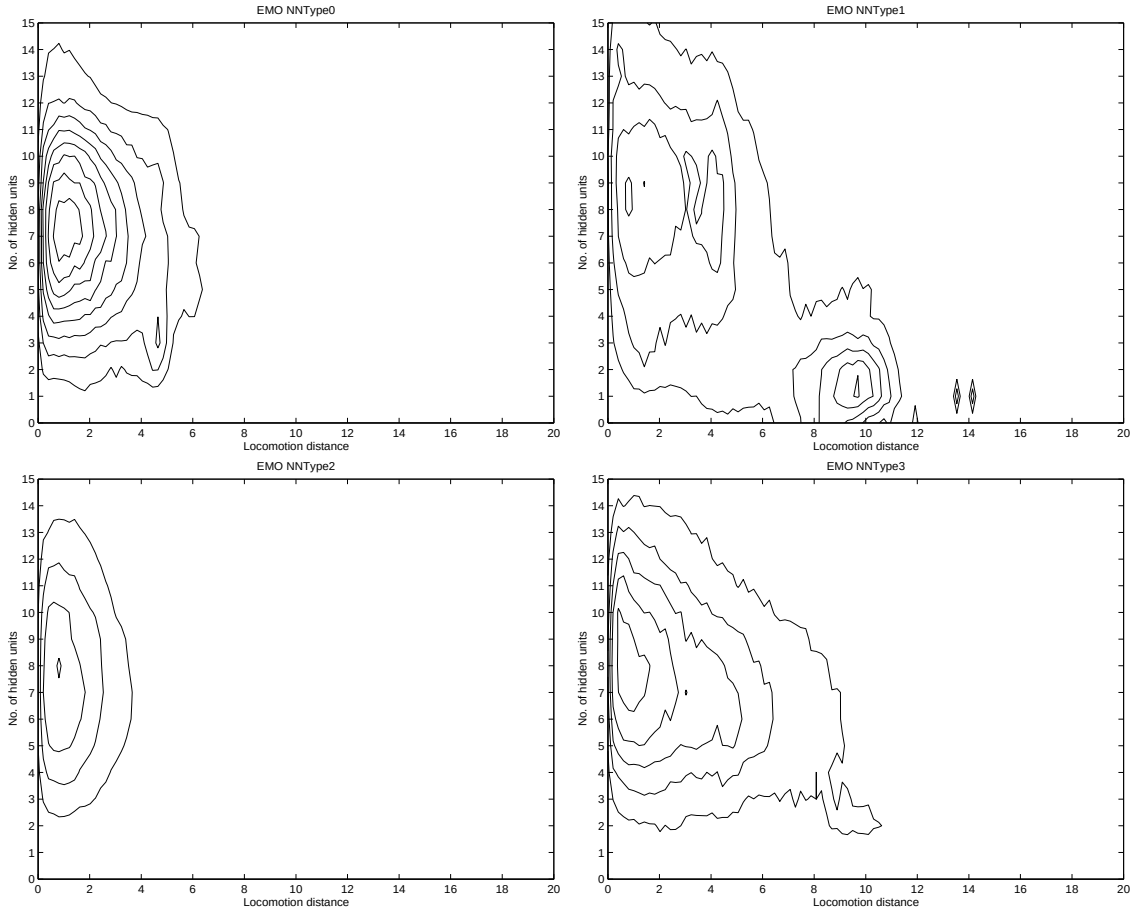


Figure 5.8: Contour graphs of frequency distribution of solutions obtained using the SPANN algorithm for ANN architecture 1. NNType0 (top left), 2. NNType1 (top right), 3. NNType2 (bottom left), 4. NNType3 (bottom right). X-axis: Locomotion distance, Y-axis: No. of hidden units.

of generating controllers approached 0 beyond a locomotion capability of 12 units whereas for NNType1 (Figure 5.9.2) and NNType3 (Figure 5.9.4), the probability only approached 0 beyond a locomotion capability of 14 units. This is another weak indication that the direct input-output connections provided an easier path to reach fitter controllers in terms of locomotion capability while the recurrent connections did not appear to provide any significant advantages over the simple feed-forward ANN architecture that had no extra connections between layers.

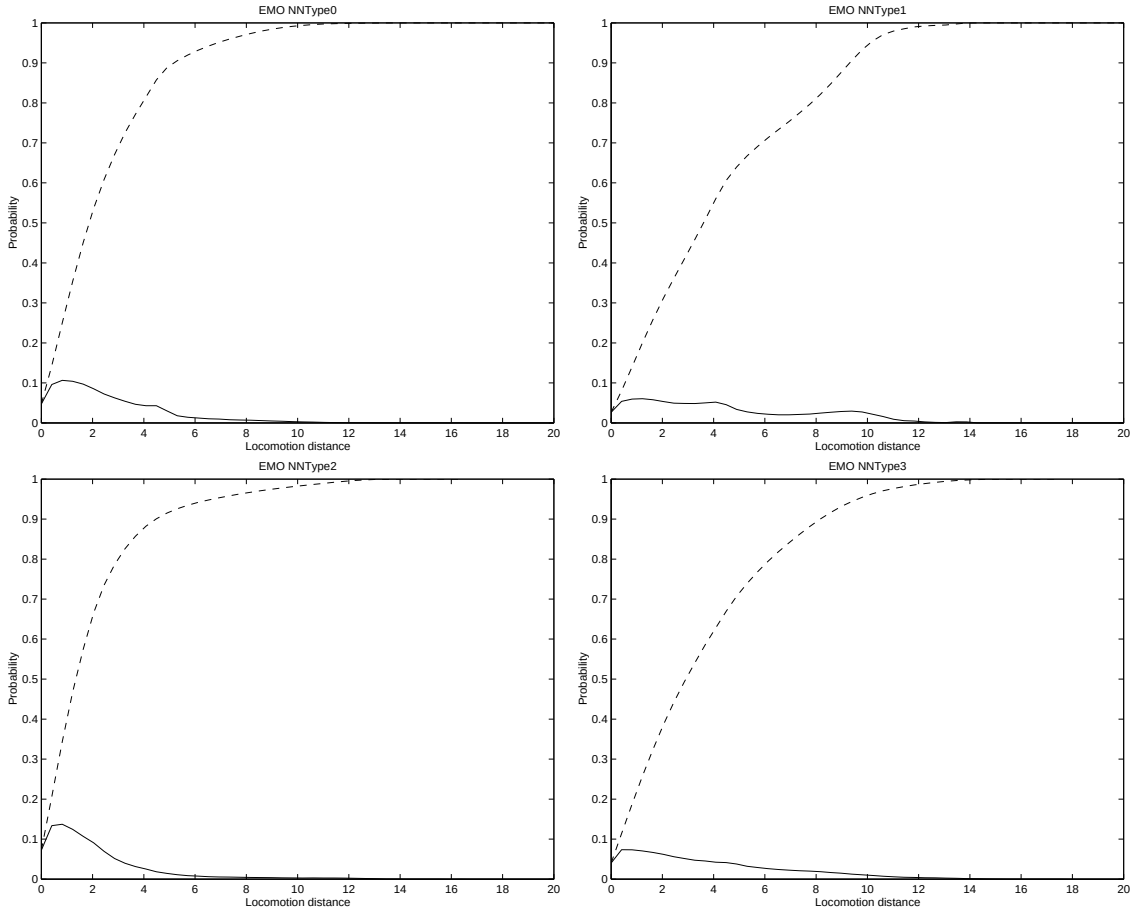


Figure 5.9: Density (solid) and cumulative (dashed) probability distribution of solutions obtained using the SPANN algorithm for ANN architecture 1. NNType0 (top left), 2. NNType1 (top right), 3. NNType2 (bottom left), 4. NNType3 (bottom right). X-axis: Locomotion distance, Y-axis: Probability.

5.7 Operational Dynamics

5.7.1 Behavior Inside and Outside Evolutionary Window

A top-down view of the artificial creature's path as controlled using the overall best ANN evolved for locomotion distance is plotted in Figure 5.10.1 for the actual period between 1–500th timestep where the fitness of the controller is being evaluated. A second graph of the artificial creature's path for the period between the 501–1000th timestep is plotted in Figure 5.10.2 to observe the locomotion be-

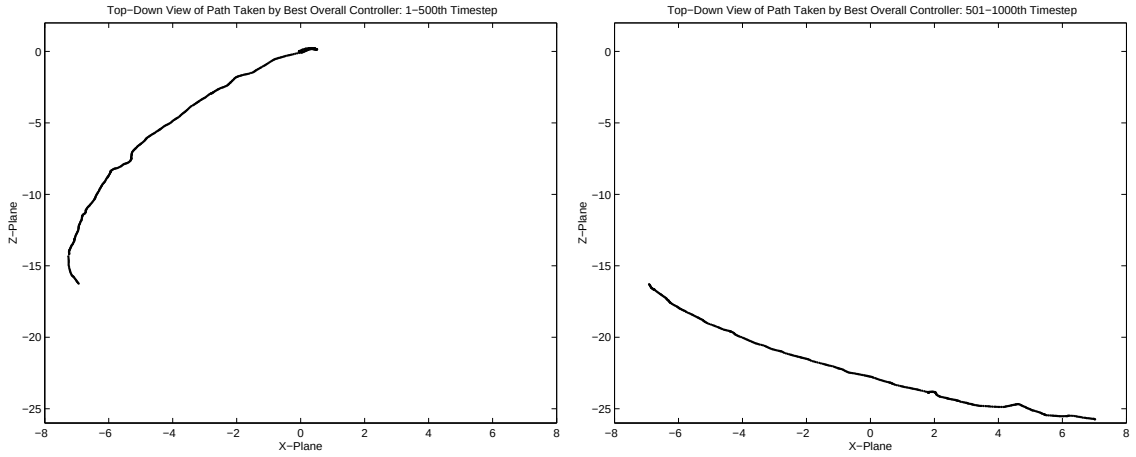


Figure 5.10: Path taken by artificial creature as controlled by overall best ANN evolved for locomotion distance using the SPANN algorithm — NNType3: 4 Hidden Units. 1. 1–500th timestep (left), 2. 501–1000th timestep (right) X-axis: X-Plane, Y-axis: Z-Plane (in Vortex, the Z-Plane is the Y-axis).

havior beyond the actual fitness evaluation window used during evolution. This controller has the NNType3 architecture and uses 4 hidden units. Although the path taken was not exactly a straight line, it did however maximize the horizontal distance moved fairly well. It begins from the origin at coordinates (0,0) and after 500 timesteps ends approximately at coordinates $(-6.9,-16.5)$. This emergent behavior is interesting in that although the initial setup has the creature's forwards orientation as being in the positive coordinate areas of the X-Plane and Z-Plane, the evolved locomotion behavior was in fact a backwards oriented walk if the initial positioning of the creature is taken as the reference frame. Visualization of the other global Pareto solutions revealed similar orientations for the evolved locomotion behaviors (interested readers can view video clips of these evolved behaviors in the accompanying CD-ROM). This can be explained by the fact that all the global Pareto solutions using the NNType3 architecture were actually obtained from one run using a particular seed. Across the global Pareto solutions obtained with other architectures using SPANN as well as other algorithms, the majority of the evolved controllers produced movement in the forwards direction rather than this backwards

movement. The design of the creature's limbs and in particular how the joint constraints were set up allowed for both forwards and backwards oriented walks to be evolved. Note that the initial movement seen in the other direction from coordinates (0,0) to approximately (0.5,0) occurred during the standing up phase of the creature's locomotion. Once the creature stood up, it then started the backwards oriented walk to achieve the maximal horizontal distance moved.

Towards the end of the walk, the path could be seen to start curving back towards the X-Plane. The continuation of this peculiar behavior beyond 500 timesteps as controlled by this evolved ANN can be seen in the plot of the path in Figure 5.10.2. Nonetheless, the creature was still able to walk in a fairly straight line thereby achieving a reasonably maximal locomotion distance during this next 500 timesteps. If the path of the creature is considered over the entire 1-1000th timestep, what this analysis shows is that the operational dynamics of the evolved behavior during the period which the controller was actually evolved to perform can be quite different to the operational dynamics when used beyond its evolutionary design period. This phenomenon relates back to what was highlighted by Ronald and Sipper (2001) in that the use of biologically-inspired solutions in engineering problems may be problematic because unexpected and sometimes unwanted results or behaviors might arise (discussed in the last paragraph of Section 2.3).

5.7.2 Limb Dynamics

The outputs generated from the operation of the overall best controller evolved for locomotion distance to Actuators y_1–y_8 are plotted in Figure 5.11. In all except one of the outputs, sine-like wave signals were generated by the evolved ANN to the motors in the respective limb actuators. This is consistent with the evolved walking behavior which is cyclic in nature.

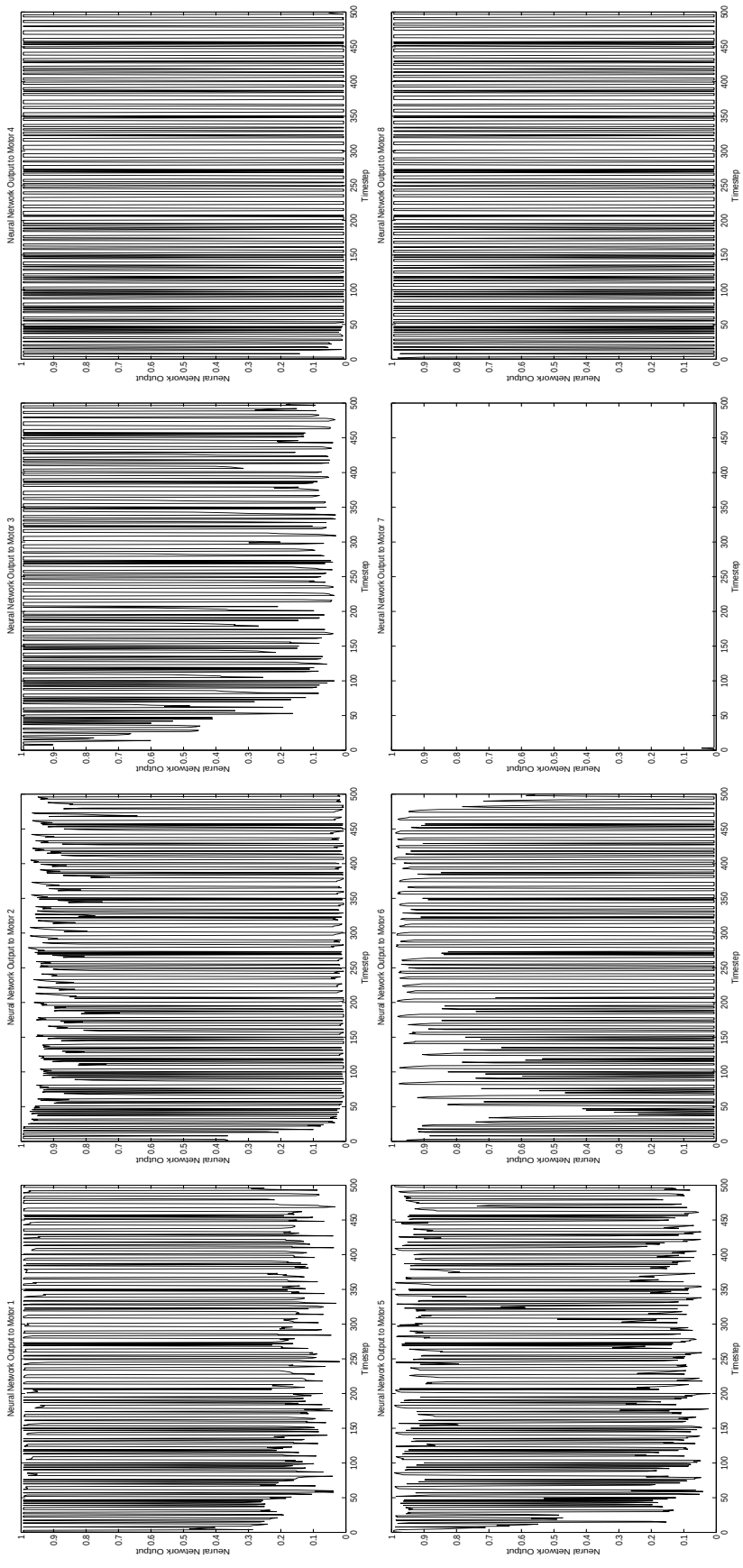


Figure 5.11: The outputs from the overall best evolved ANN controller to Actuator 1. y_1 (top extreme left), 2. y_2 (top middle left), 3. y_3 (top middle right), 4. y_4 (top extreme right), 5. y_5 (bottom extreme left), 6. y_6 (bottom middle left), 7. y_7 (bottom middle right), 8. y_8 (bottom extreme right). X-axis: Timestep, Y-axis: Neural network output.

It is also interesting to note that the signals generated were quite distinctive over time as near maximal outputs of either 0 and 1 at the peaks and troughs of the cycle were being generated by the ANN. This behavior is close to the optimal control strategy known as "Bang-Bang Controls", which take their values on the extreme points (in our case, 0 and 1) (Macki and Strauss 1982). In terms of the single output which generated a totally flat signal of practically 0 magnitude over time (Figure 5.11.7), this indicated the presence of a passively-controlled lower limb, which obtained its swinging motion from the movement of the attached upper limb. A visual inspection of the creature in simulation confirmed that this limb did in fact exhibit some dynamical behavior during locomotion. This suggests that the evolutionary search found a simpler control solution through the use of a passive dynamic limb (McGeer 1990).

Limbs	Correlation Coefficient
Upper and Lower Back Left	0.9410
Upper and Lower Front Left	−0.9648
Upper and Lower Back Right	0.0376
Upper and Lower Front Right	−0.9998

Table 5.6: Correlation coefficients for neural network outputs between the upper and lower limbs of each leg.

Another interesting dynamical behavior that emerged from the outputs of the ANN is that the component limbs in all of the legs learnt to coordinate and synchronize their movements within each leg, with the exception of the leg containing the passive dynamic limb. This is evidenced by the very high correlation between the upper and lower limbs of the back left, front left and front right legs as shown in Table 5.6. For a legged gait with good locomotion capabilities, the constituent components in each leg would be expected to function as a cohesive unit in order for each leg to generate useful movements for locomotion. The outputs to the individual limb actuators for the back left leg have evolved to be almost entirely in-phase, as evidenced by the very high positive correlation coefficient (0.9419). On the other hand, the outputs to the individual limb actuators for the front left and front right legs have evolved to be almost entirely out-of-phase, as evidenced by the very high

negative correlation coefficients (-0.9648 & -0.9998). This shows that the neural network controller has learnt to coordinate and synchronize between the limbs of each leg, thereby allowing for successful locomotion to occur.

5.7.3 Effects of Noise

In this section, we investigate the effects of noise on the performance of the overall best evolved controller for locomotion distance obtained from SPANN. Seven different levels of noise were applied to the joint angle sensors, touch sensors and outputs to the actuators individually as well as in combination for all three elements. Random noise levels ranging from 1% to 50% of the individual ranges of values for these sensors and actuators (see Section 3.2) were applied.

Noise Level	Locomotion Distance with Noise in Joint Angle Sensors	Locomotion Distance with Noise in Touch Sensors	Locomotion Distance with Noise in Actuators	Locomotion Distance with Noise in All Sensors and Actuators
1%	13.4402 ± 1.7281	14.3599 ± 0.9073	13.8640 ± 0.9970	13.8147 ± 1.2219
5%	13.2121 ± 2.0271	12.5348 ± 1.0275	13.2310 ± 0.7213	11.2828 ± 0.8163
10%	11.8406 ± 1.2960	10.6214 ± 1.3914	12.9669 ± 1.2229	9.5492 ± 0.5552
20%	8.7969 ± 0.8017	6.3701 ± 1.9298	10.8933 ± 2.5271	4.8763 ± 0.4763
30%	6.8828 ± 0.4429	3.5598 ± 1.2999	8.8441 ± 1.9260	2.6185 ± 1.0707
40%	6.6890 ± 0.6028	1.6049 ± 1.1109	6.7339 ± 1.7209	1.2486 ± 0.6111
50%	6.7256 ± 0.9367	1.3050 ± 1.0950	3.7132 ± 1.9092	1.2153 ± 0.4210

Table 5.7: Comparison of average locomotion distance achieved over 10 runs by overall best controller evolved for locomotion distance using the SPANN algorithm with varying noise levels in the sensors and actuators.

Table 5.7 lists the average and standard deviation of locomotion distances achieved by the overall best locomotion controller from SPANN with varying levels of noise applied to the sensors and actuators of the artificial creature. The performance of the controller degraded monotonically in all cases as the level of noise was increased from 1% to 50%, except for 50% noise in the joint angle sensor. At the lowest level of noise of 1%, the least significantly affected component was the touch sensor which still achieved on average 81.1% of the original locomotion distance.

However, as the noise level was increased, the touch sensors seemed to be most affected by the presence of noise. In all cases where noise ranging from 5% to 50% was applied to the individual components, the lowest average locomotion distance was obtained when there was the presence of noise in the touch sensors. When noise was introduced to all sensors and actuators of the artificial creature in combination, the performance was lower than in all cases where noise was applied to each individual component except for 1% noise in the joint angle sensors. A visual inspection of the artificial creature in simulation with 10% random noise added to the sensors and actuators both individually and in combination revealed that the major characteristics of the locomotion behavior was still present, such as the backwards oriented walk and general movement of the limbs (interested readers can view video clips of a sample of these behaviors in the accompanying CD-ROM). Therefore, the evolved controller was still able to perform reasonably well with low levels of noise present in the sensors and actuators of the artificial creature.

5.8 Advantages of Pareto EMO

From these experiments, it can be seen that the most significant advantage in using a Pareto EMO approach for studying the evolution of artificial creatures as proposed in our SPANN algorithm is that a variety of controllers can be generated in a single evolutionary run without requiring any further modification of parameters by the user. This means that an entire set of controllers with varying network sizes and locomotion capabilities can be generated at once, allowing for comparisons between creatures with different abilities and controllers to be made after just a single run is conducted for each type of creature. As such, the Pareto EMO approach provides a flexible and convenient platform for conducting investigations into the evolution of artificial creatures. This represents a significant advantage over single-objective evolutionary systems that need to be re-run multiple times in order to test the effect of other factors such as number of hidden units on the locomotion capability of the artificial creatures (Bongard and Pfeifer 2002). Such a setup would require a

significantly larger number of evolutionary runs before a suitable set of controllers with different network characteristics and locomotion capabilities can be obtained in order to conduct comparisons between different creature designs. An alternative method would be to re-formulate the problem by taking a weighted sum of the two objectives into a single objective. However, there are a number of drawbacks in using a weighted sum methodology, which we discuss in detail later in Section 6.4.2. In short, the Pareto EMO paradigm allows the user the option to conveniently choose from a variety of controllers with varying architectural complexities and locomotion competencies to suit the eventual simulation environment, constraints and purposes.

A further advantage of using a Pareto multi-objective approach for artificial evolution is that genetic diversity is maintained naturally during the course of the evolutionary process (Abbass and Deb 2003). A common problem with evolutionary optimization algorithms is premature convergence due to loss of genetic diversity and this phenomenon has been observed to cause problems in the artificial evolution of virtual creatures as well (Komosinski and Rotaru-Varga 2001). In a simple artificial life ecosystem, mutualism (Pachepsky, Taylor, and Jones 2002) was proposed as a method for promoting genetic diversity and was shown to improve evolvability as well as population stability in the artificial evolutionary system. Here, we propose an evolutionary multi-objective algorithm that promotes reproductive diversity by allowing the evolutionary process to optimize along two separate and distinct goals of minimizing network size while maximizing locomotion ability. This type of evolutionary optimization algorithm therefore fits well into the scheme of creating artificial creatures. In this case, evolutionary creativity should not be stifled by the optimization process but should instead be encouraged if interesting and diverse creatures are to emerge from the process.

The use of EMO also opens up the possibility of creating extra-dimensional bypasses through the search space that provide an easier path for the optimization process to reach fitter regions of the solution space. This phenomenon has been previously encountered in conventional single-objective evolutionary optimization systems through the addition of extra genotypic parameters into the artificial evo-

lutionary system (Bongard and Paul 2001). However, such artificial methods of increasing the search space is rather subjective and may require significant experimentation in order to find the right parameters that can create the extra-dimensional bypass. In our proposed methodology, this is achieved in a natural manner through the use of multiple optimization objectives. Here, we do not make the claim that such extra-dimensional bypasses will be present under all EMO runs but simply that the use of multiple objectives in an artificial evolutionary system allows for the possibility of such bypasses to emerge through the inherently more complex interactions between genes under multiple evolutionary pressures. The presence of high epistatis is evidenced by the highly rugged fitness landscape areas present in certain sub-regions of the search space as discussed in Section 4.5.3.1.

5.9 Chapter Summary

An investigation into the use of multi-objective evolutionary optimization for automatic synthesis of ANN controllers that are proficient at generating locomotion capabilities in a physically simulated quadruped yielded the following results:

- An EMO algorithm called SPANN was implemented for the multi-objective evolution of artificial creature controllers. ANN controllers based on four different types of underlying architecture were successfully evolved for maximum horizontal locomotion capability and minimum usage of number of hidden units in the ANN.

- An EMO algorithm provides significant advantages over conventional single-objective optimization algorithms by: (1) reducing the number of runs required to test different design factors associated with the synthesis of artificial creatures, (2) preserving genetic diversity and, (3) offering extra-dimensional bypasses for the search process to reach fitter solution spaces.

- Additional recurrent connections do not provide any significant advantages over conventional feed-forward neural network architectures for evolving loco-

motion capability in a four-legged artificial creature.

- Pure reactive agents not requiring hidden layer transformations in the ANN controller produced sufficiently good locomotion capabilities. The use of direct input-output connections in a perceptron-like controller was sufficient for generating a basic locomotion ability in a four-legged artificial creature. The use of a hidden layer was not required to synthesize a controller with a comparatively good locomotion capability.

- An operational dynamics analysis revealed that the ANN controller learnt to coordinate and synchronize between the upper and lower limbs in three of the simulated quadruped's legs. The presence of a passively-controlled dynamic limb was observed in the remaining leg. The ANN controller still performed well when a reasonable level of noise was present in the sensors and actuators.

The design, implementation and use of an EMO algorithm called SPANN for the evolution of artificial creature controllers has been presented in this chapter. ANN controllers were successfully evolved for minimum hidden layer size and maximum horizontal locomotion distance using four different types of ANN architecture. In the next chapter, we will compare the SPANN algorithm against more conventional methods of evolutionary optimization to verify that this approach is actually beneficial for evolving artificial creature controllers.

Chapter 6

Verifying the Self-Adaptive Pareto EMO

In this chapter, we employ more conventional methods of evolutionary optimization for the generation of artificial creature controllers and then compare these results against the self-adaptive Pareto approach of the SPANN EMO algorithm. Three evolutionary optimization algorithms are used here, namely a hand-tuned EMO algorithm, a weighted sum EMO algorithm and a single-objective evolutionary optimization algorithm. The objectives of these comparisons are firstly to elucidate the effectiveness of using these conventional algorithms for generating high quality locomotion controllers and secondly whether the advantages of the self-adaptive Pareto approach are truly beneficial against these more common methods of evolutionary optimization. A test of redundancy present in the ANN controllers evolved using SPANN is also conducted and compared against the ANN controllers evolved using the conventional EAs mentioned above. Finally, a comparison of SPANN is made against NSGA-II, a well-known and state-of-the-art Pareto EMO algorithm.

6.1 A Hand-Tuned EMO Algorithm

6.1.1 Experimental Setup

In this set of experiments, we used an EMO algorithm with user-defined crossover and mutation rates rather than self-adapting parameters in the SPANN algorithm (similar to the MPANN algorithm (Abbass 2001) but without back-propagation). Apart from the non-self-adapting crossover and mutation rates, the hand-tuned EMO algorithm is otherwise similar to the SPANN algorithm in all other respects. The NNType3 architecture was used since it provided the best overall results among the different controller architectures (see Section 5.6). Three different crossover rates **(c)** and mutation rates **(m)** were used: 10%, 50% and 90% for both rates giving a total of 9 different combinations. As with SPANN, the fitness of each genotype in these experiments was evaluated according to both the f_1 and f_2 objective functions, which measures the locomotion distance achieved and number of hidden units used by the controller respectively as defined in Section 3.4.1. All other evolutionary and simulation parameters remain the same: 1000 generations, 30 individuals, maximum of 15 hidden units, 500 timesteps and 10 repeated runs. In characterizing the fitness landscapes, the individual genotypes were grouped into 6250 discrete intervals over the increased locomotion distance dimension of 25 used in this chapter, in order to maintain the same frequency distribution's interval length as those previously used in Chapters 4 and 5.

6.1.2 Results and Discussion

In this section, we discuss the solutions produced using the hand-tuned EMO algorithm in terms of the different crossover and mutation rates used during evolution. The best Pareto solutions for locomotion distance obtained from conducting the evolutionary optimization process using user defined rates for the genetic operators are presented in Table 6.1. The highest overall best locomotion distance of 19.5 was achieved using a crossover rate of 10% and a mutation rate of 50% while the lowest overall best locomotion distance of 14.9 was obtained using

Crossover Rate	Mutation Rate	Overall Best Locomotion Distance	Average Best Locomotion Distance ± Standard Deviation	No. of Hidden Units
10%	10%	17.1071	13.5192 ± 2.7845	3.2 ± 1.8
10%	50%	19.5051	14.1158 ± 2.6535	6.6 ± 2.1
10%	90%	14.9493	13.2843 ± 1.8225	8.1 ± 2.7
50%	10%	16.2272	14.1268 ± 2.1286	3.0 ± 2.4
50%	50%	18.5638	15.3819 ± 2.3195	6.5 ± 1.7
50%	90%	16.3347	13.1881 ± 1.4715	7.7 ± 2.5
90%	10%	18.5980	14.1511 ± 2.4721	3.4 ± 1.8
90%	50%	15.8766	13.1978 ± 1.6447	6.1 ± 1.4
90%	90%	15.9395	12.1653 ± 2.3799	6.8 ± 2.9

Table 6.1: Comparison of best locomotion distance for Pareto solutions found over 10 independent runs using the hand-tuned EMO algorithm with different crossover and mutation rates.

a crossover rate of 10% and a mutation rate of 90%. The best result in terms of average best locomotion distance achieved was obtained using a crossover rate of 50% and mutation rate of 50% while the worst overall result was obtained using a crossover rate of 90% and mutation rate of 90%. This suggests that a low to medium crossover coupled with a medium mutation rate provided better results when self-adaptation was not used in the EMO algorithm and conversely, a high mutation rate seemed to provide lower quality results for locomotion distance. In terms of optimizing the hidden layer, the crossover rate did not seem to affect the results while a low mutation rate of 10% consistently gave the smallest hidden layer for the evolved controllers of around 3.2 hidden units compared to higher mutation rates of 50% and 90%.

The evolution of the Pareto solution for best locomotion distance using the hand-tuned EMO algorithm for 10 runs over 1000 generations is shown in Figure 6.1 for four of the nine different combinations of crossover and mutation rates. In general, the characteristics of these convergence graphs were similar to that obtained using SPANN (Figure 5.3.4) in that the progression of fitter solutions being discovered was relatively smooth over time. The top two graphs (Figures 6.1.1 & 6.1.2) are representative of these runs. However, there were also combinations of crossover and

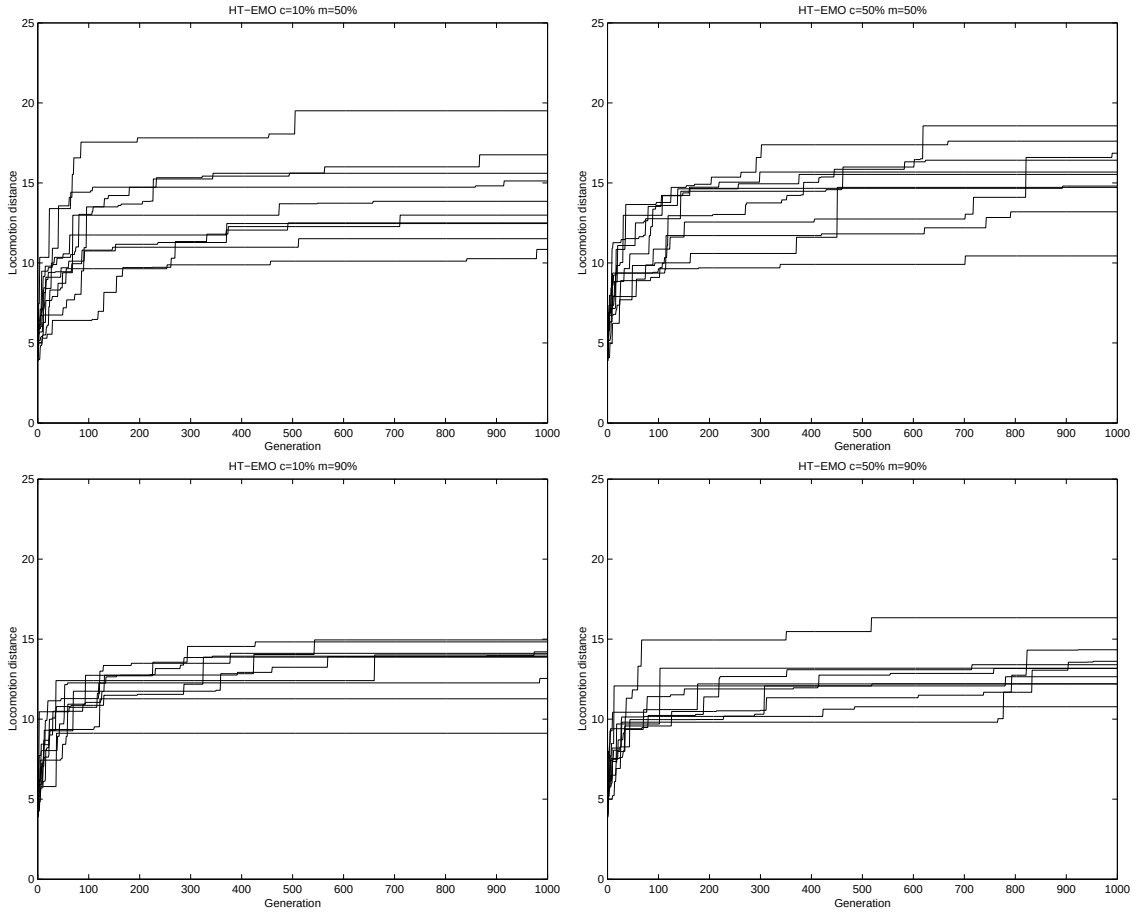


Figure 6.1: Best locomotion distance of Pareto solutions obtained over 1000 generations for 10 runs using the hand-tuned EMO algorithm with 1. c=10% m=50% (top left), 2. c=50% m=50% (top right), 3. c=10% m=90% (bottom left), 4. c=50% m=90% (bottom right). X-axis: Generation, Y-axis: Locomotion distance. Additional graphs can be found in the accompanying CD-ROM.

mutation rates where the improvement of the solutions was much less smooth causing large plateau regions. This phenomenon can be seen in the bottom two graphs (Figures 6.1.3 & 6.1.4), which are highly reminiscent of the graph obtained using hill-climbing (Figure 4.8.4), suggesting that the algorithm may have become stuck in a local optimum for some of the runs. This is likely to be due to the limitation of not being able to change the mutation and crossover rates during the evolutionary optimization process, which may be beneficial in escaping from deep local optima.

As such, these results suggest that self-adaptation, such as that present in SPANN, may be a desirable feature not merely for reducing the computational runs required to test out hand-tuned crossover and mutation rates but also for avoiding premature convergence to less optimal solutions.

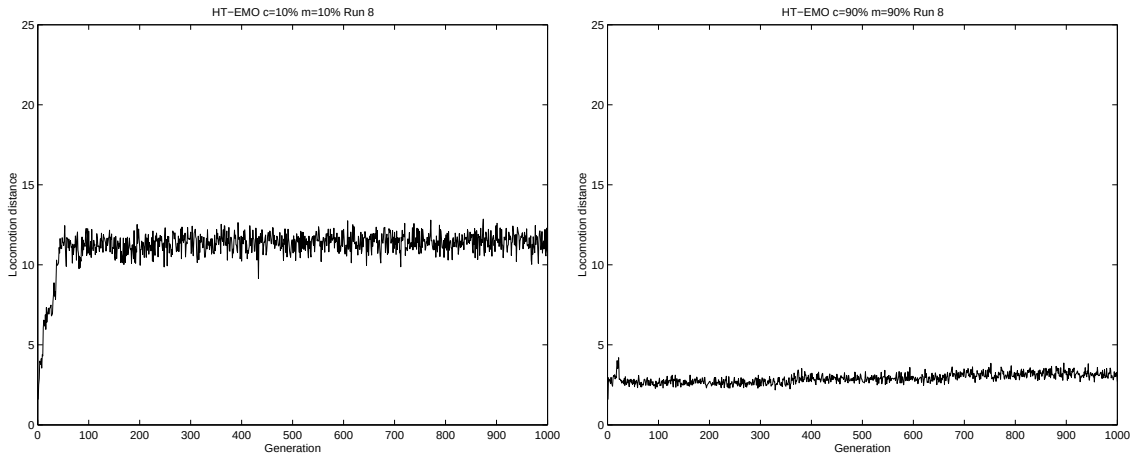


Figure 6.2: Mean locomotion distance of population over 1000 generations using the eighth seed for the hand-tuned EMO algorithm with 1. c=10% m=10% (left), 2. c=90% m=90% (right). X-axis: Generation, Y-Axis: Locomotion distance. Additional graphs can be found in the accompanying CD-ROM.

Figure 6.2 depicts the mean locomotion distance and Figure 6.3 depicts the standard deviation for locomotion distance of the population as it evolved over 1000 generations. The graphs depicted are representative of the trends observed in a large majority of the runs. The average population fitness in terms of locomotion distance remained constant within a fixed range after the initial large jump early during evolution as shown in Figures 6.2.1 and 6.2.2. The two most common trends in the standard deviation of the population were remaining fairly constant within a certain range as shown in Figure 6.3.1 and increasing slightly over time as shown in Figure 6.3.2. Lower crossover and mutation rates seemed to produce a higher population average (Figure 6.2.1) compared to higher crossover and mutation rates (Figure 6.2.2). On the other hand, higher crossover and mutation rates appeared to produce less changes to the standard deviation of the population over time (Figure

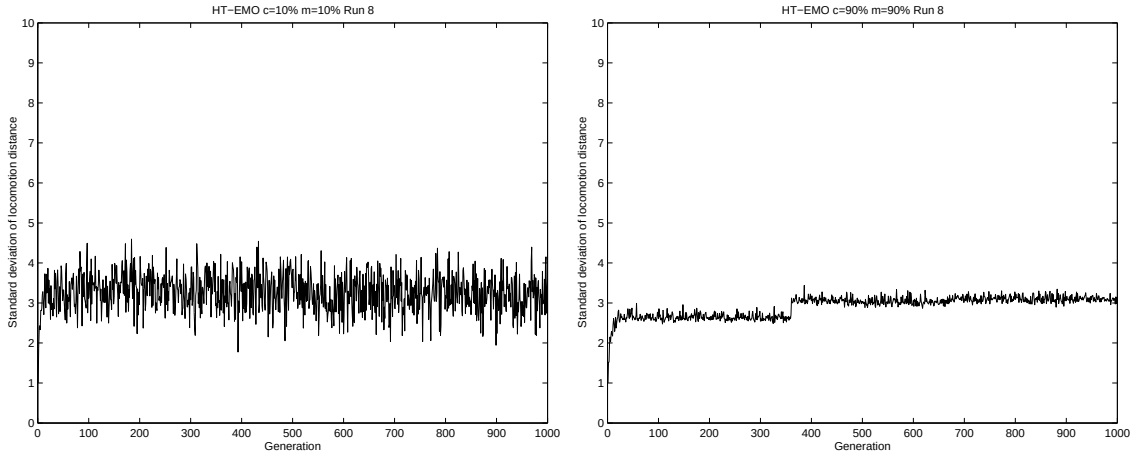


Figure 6.3: Standard deviation for locomotion distance of population over 1000 generations using the eighth seed for the hand-tuned EMO algorithm with 1. c=10% m=10% (left), 2. c=90% m=90% (right). X-axis: Generation, Y-Axis: Standard deviation of locomotion distance. Additional graphs can be found in the accompanying CD-ROM.

6.3.2) compared to lower crossover and mutation rates (Figure 6.3.1). These last two observations suggest that higher crossover and mutation rates were less efficient at finding fitter solutions because the large changes being applied to the genotype at every generation did not allow evolution a chance to discover and maintain a good set of basic genes. Consequently, the solutions being sampled using high crossover and mutation rates were similarly low in fitness resulting in the lower population means and smaller changes in the standard deviations over time.

6.1.3 Search Space Characterization

The distribution of genotypes generated using the hand-tuned EMO algorithm is plotted in Figure 6.4 in terms of locomotion distance and number of hidden units used in the ANN. Although the distribution of solutions across both objective spaces were more uniform compared to that obtained using random search (Figure 4.1.4), hill-climbing (Figure 4.5.4) and random walk (Figure 4.9.4), they were much less uniform and more clustered compared to the distribution obtained using

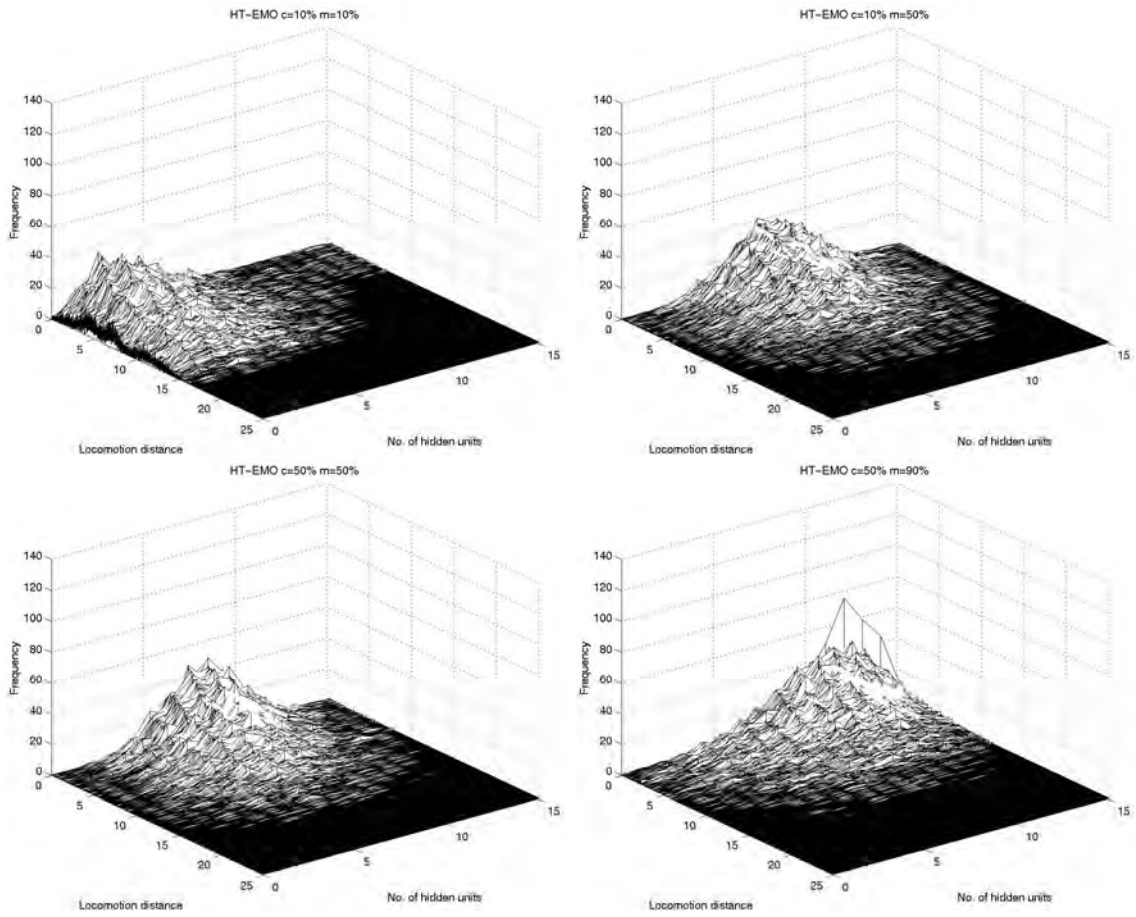


Figure 6.4: Frequency distribution of solutions obtained using the hand-tuned EMO algorithm with 1. c=10% m=10% (top left), 2. c=10% m=50% (top right), 3. c=50% m=50% (bottom left), 4. c=50% m=90% (bottom right). X-axis: Locomotion distance, Y-axis: No. of hidden units, Z-axis: Frequency. Additional graphs can be found in the accompanying CD-ROM.

SPANN (Figure 5.6.4). From the general features observed in these graphs, the distributions most similar to SPANN were those generated using a crossover rate of 10% and mutation rate of 50% (Figure 6.4.2) as well as a crossover rate of 50% and mutation rate of 50% (Figure 6.4.3). The main difference between the hand-tuned EMO algorithm and SPANN is that the solutions in the hand-tuned EMO were mainly clustered around the lower fitness regions of the search space, a large majority of which yielded only between 0 and 10 units of locomotion distance compared

to SPANN where a more even distribution could be seen to extend to fitness regions of 15 units of locomotion distance.

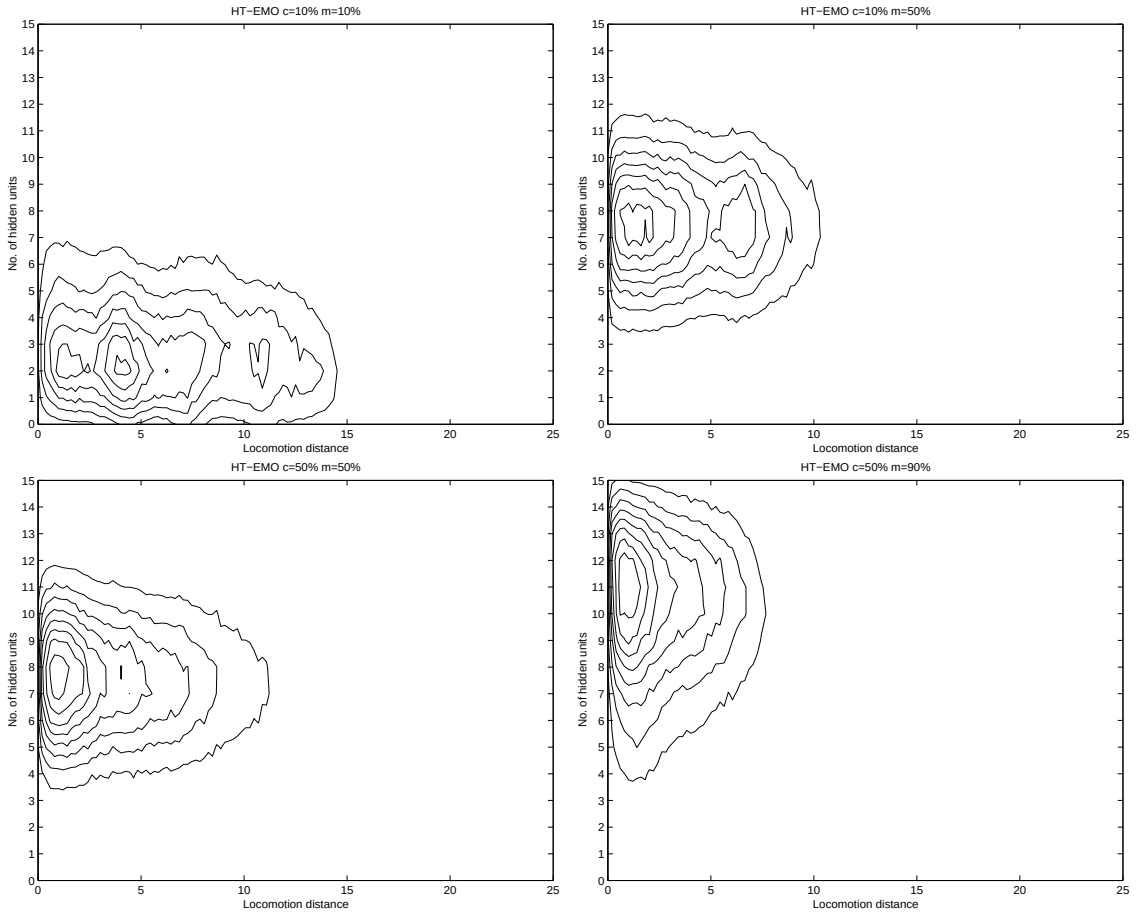


Figure 6.5: Contour graphs of frequency distribution of solutions obtained using the hand-tuned EMO algorithm with 1. c=10% m=10% (top left), 2. c=10% m=50% (top right), 3. c=50% m=50% (bottom left), 4. c=50% m=90% (bottom right). X-axis: Locomotion distance, Y-axis: No. of hidden units. Additional graphs can be found in the accompanying CD-ROM.

The contour graphs in Figure 6.5 illustrate the distribution of solutions across the two objectives of minimizing the hidden layer and maximizing locomotion distance. A number of interesting features emerged in these contour graphs. Firstly, the mutation rate significantly affected the range of genotypes generated in terms of the number of hidden units used in the controller. The controllers evolved using

the lowest mutation rate of 10% centered around a usage of between 2 and 3 hidden units (Figure 6.5.1). When the mutation rate was increased to 50%, the solutions now centered around a higher usage of between 7 and 8 hidden units (Figures 6.5.2 & 6.5.3), and furthermore, in the highest setting of the mutation rate at 90%, the solutions clustered around controllers that used between 10 and 12 hidden units (Figure 6.5.4). The contour features most similar to SPANN (Figure 5.8.4) again could be seen when the crossover rate was set at 10% and mutation rate at 50% (Figure 6.5.2) as well as at a crossover rate of 50% and mutation rate of 50% (Figure 6.5.3), although in both these cases the spread of solutions over the objective spaces were less uniformly distributed. Interestingly, the movement of solutions to larger hidden layer sizes produced lower locomotion capabilities. The underlying fitness landscape may have become more rugged as the size of hidden layer increased and as previously postulated, the non-self-adapting crossover and mutation rates may have represented a severe limitation in allowing the algorithm to move through these landscapes, subsequently causing the optimization process to become trapped around sub-optimal regions of the search space.

The probability density function of solutions obtained using the hand-tuned EMO algorithm is illustrated in Figure 6.6. The probability density curves show that a low crossover rate of 10% and low mutation rate of 10% provided the best distribution of solutions across the locomotion objective space (Figure 6.6.1). The cumulative curve shows that the probability of encountering fitter solutions decreased noticeably as the mutation rate was increased to 50% (Figures 6.6.2 & 6.6.3) and especially 90% (Figure 6.6.4), where the probability of encountering a solution decreased to 0 beyond a fitness of only around 11.

The search space characterization of the hand-tuned EMO algorithm showed significantly different characteristics compared to SPANN, especially when very high and very low combinations of crossover and mutation rates were used. As previously discussed, the correct choice of these two rates by the user is paramount in obtaining reasonably good results when using this form of the EMO algorithm. This analysis also showed that the use of self-adaptive crossover and mutation rates

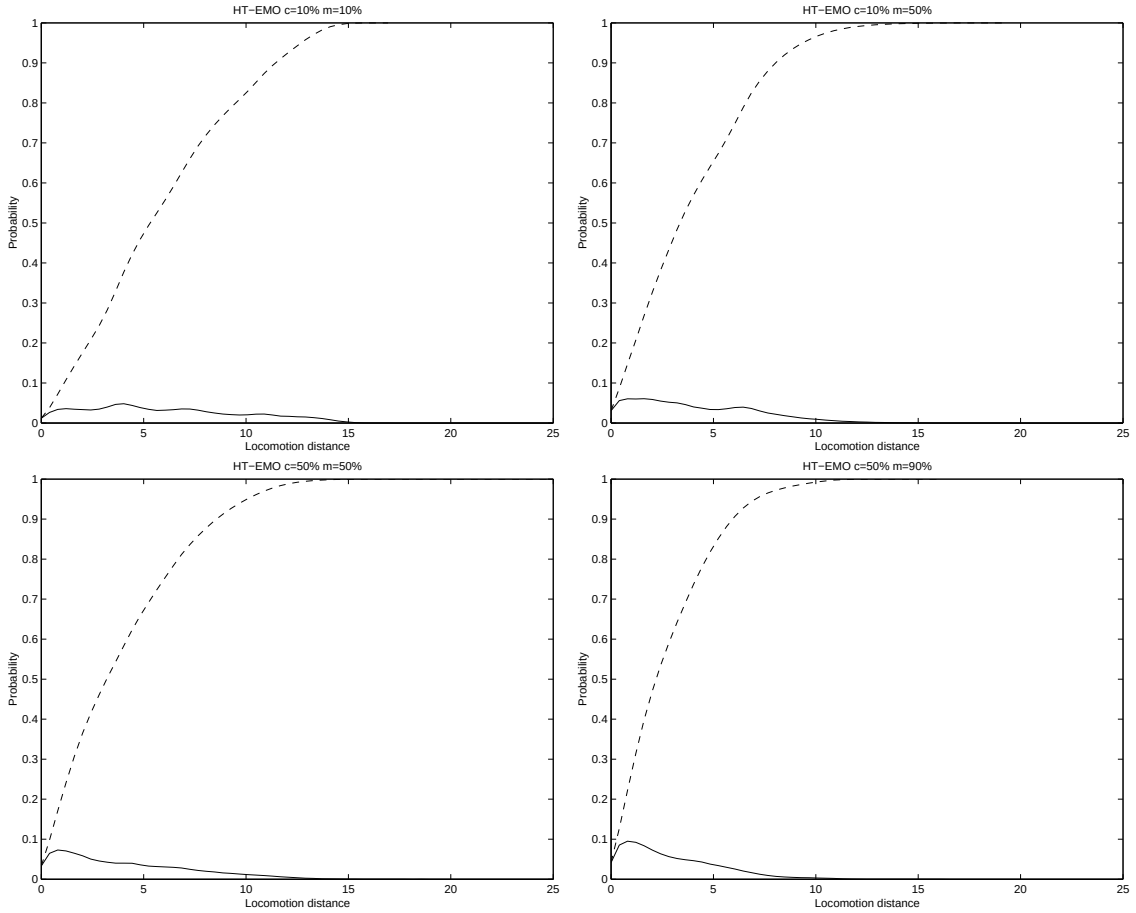


Figure 6.6: Density (solid) and cumulative (dashed) probability distribution of solutions obtained using the hand-tuned EMO algorithm with 1. c=10% m=10% (top left), 2. c=10% m=50% (top right), 3. c=50% m=50% (bottom left), 4. c=50% m=90% (bottom right). X-axis: Locomotion distance, Y-axis: Probability. Additional graphs can be found in the accompanying CD-ROM.

in SPANN allowed the evolutionary search to sample a much larger area of the objective space compared to the non-self-adaptive EMO algorithm and as such was able to perform more effectively in finding good locomotion controllers.

6.2 A Weighted Sum EMO Algorithm

6.2.1 Experimental Setup

For this second set of experiments, we used a single objective that combined the two objectives f_1 and f_2 using a weighted sum rather than a true Pareto approach as found in the SPANN algorithm, which distinctly separates the two objectives when assigning fitness values to individuals in the population. As in the comparison between the hand-tuned versus self-adaptive EMO algorithms in the previous section, the NNType3 architecture was used in this set of experiments for the same reason that it provided the best overall result among the different architectures as reported in Section 5.6. The weighting of the individual objectives was done in a relative manner using a parameter denoted by γ. In order to combine the two objectives f_1, which is the maximization of the locomotion distance, and f_2, which is the minimization of the number of hidden units used in the ANN, into a single weighted sum fitness function, these objectives needed to be unified in terms of their direction of optimization. Firstly, the locomotion distance objective f_1 was re-defined to be a minimization problem

$$f_1' = 100.0 - f_1 \tag{6.1}$$

which yielded the minimization of the overall weighted sum function as follows:

$$\gamma \times f_1' + (1 - \gamma) \times f_2 \tag{6.2}$$

However, we chose to convert the overall weighted sum optimization problem into one of maximization to maintain consistency when presenting the solutions in terms of locomotion distance achieved by the best evolved controllers. Hence the final weighted sum objective function is given by

$$f(overall) = 100.0 - [(\gamma \times f_1') + ((1 - \gamma) \times f_2)] \tag{6.3}$$

where $f(overall)$ represents the weighted fitness and γ is the relative weight parameter. The unification of the two objectives f_1 and f_2 could have been similarly

achieved by converting f_2 into a maximization problem. 10 different values were used for γ ranging from 10% to 100% in increments of 10%. No setup for $\gamma = 0\%$ was used since in this case, no optimization would be performed on f_1, which means that in such an evolutionary run, there will be no pressure for the controller to develop any locomotion ability whatsoever. On the other hand, $\gamma = 100\%$ would result in an evolutionary run with no pressure towards minimizing the number of hidden units since no optimization would be performed on f_2. All this means is that the hidden layer in the controller is completely free to use any number of hidden units for optimizing the locomotion behavior, therefore this setup is retained in the experiments.

An approach similar to the $(\mu + \lambda)$ evolutionary strategy is used where the 15 best individuals of the population are carried over to the next generation without any modification to the genotype at all. This is to allow a setup similar to SPANN, where the upper bound on the number of Pareto solutions is simply $1 + 15$, the maximum number of hidden units allowed. The crossover and mutation operators function as in SPANN and the rates for these genetic operators are also self-adaptive. The only real difference between SPANN and the weighted sum method is the objective function and therefore, the selection mechanism. For all other parameters, they remain the same as in all other experiments: 1000 generations, 30 individuals, maximum of 15 hidden units, 500 timesteps and 10 repeated runs. As in the hand-tuned EMO algorithm search space characterization, the individual genotypes generated were grouped into 6250 discrete intervals to cater for the increased locomotion distance dimension.

6.2.2 Results and Discussion

The results obtained from using the weighted sum method for conducting the EMO process are given in Table 6.2. In the analysis of this set of results, we first present the solution fitness in terms of the actual weighted sum value that is used for the selection process. Then, we decompose the weighted sum fitness into the two separate objectives of locomotion distance and number of hidden units, and

γ	Weighted Sum Value ± Standard Deviation	Overall Best Locomotion Distance	Locomotion Distance ± Standard Deviation	No. of Hidden Units
10%	90.9857 ± 0.1428	12.3513	9.8571 ± 1.4277	0.0 ± 0.0
20%	82.0122 ± 0.4539	14.5964	10.4613 ± 2.6883	0.1 ± 0.3
30%	72.4592 ± 0.3774	9.8821	8.4306 ± 1.3288	0.1 ± 0.3
40%	63.4604 ± 0.6815	12.3985	9.4011 ± 2.1017	0.5 ± 0.8
50%	55.2962 ± 1.4375	15.8411	11.3924 ± 3.0330	0.8 ± 0.9
60%	46.6677 ± 1.9308	16.4046	12.1794 ± 2.9865	1.6 ± 1.0
70%	38.6314 ± 1.2593	17.9004	13.7448 ± 2.4376	3.3 ± 1.9
80%	30.4217 ± 1.6079	17.7011	14.0521 ± 2.3034	4.1 ± 1.5
90%	23.0408 ± 1.8146	18.1530	15.1119 ± 1.9977	5.6 ± 1.9
100%	15.2829 ± 3.6578	21.8228	15.2829 ± 3.6578	8.1 ± 1.5

Table 6.2: Best solutions obtained over 10 independent runs using the weighted sum EMO algorithm with different weights for the two objectives. γ = relative weight parameter.

present the analysis from these distinct points of view. Firstly, it is apparent that the correct combination of weights played a critical role in obtaining good results from the evolutionary optimization runs when both locomotion distance and hidden layer size are to be considered simultaneously in a single weighted objective. Setting γ to between 60% and 90% seemed to provide a good trade-off between achieving a reasonably good locomotion capability and relatively small hidden layer size. Setting $\gamma = 100\%$, which places all the optimization pressure on the locomotion component, yielded the highest overall and average best locomotion distance. However, this correspondingly resulted in the highest average of hidden units used in the evolved ANNs since there was no pressure to minimize the hidden layer at all. On the other hand, the lowest average of hidden units used was obtained when γ was set to 10%, although the average best locomotion distance achieved was much lower at only around 9.9 by virtue of the very large weighting assigned to the hidden unit component and correspondingly small weighting assigned to the locomotion component. Although it was expected that the average best locomotion distance obtained would increase monotonically as γ increased, this was not the case in the runs where the parameter was set to 30% and 40%. This was likely due to the fact that controllers with very small hidden layer sizes dominated the elite solutions and

subsequently caused the optimization process to become stuck in a local optima centered around these small-sized controllers with limited locomotion capabilities. Further analysis of this phenomenon is given in the following paragraph as well as in Section 6.2.3.

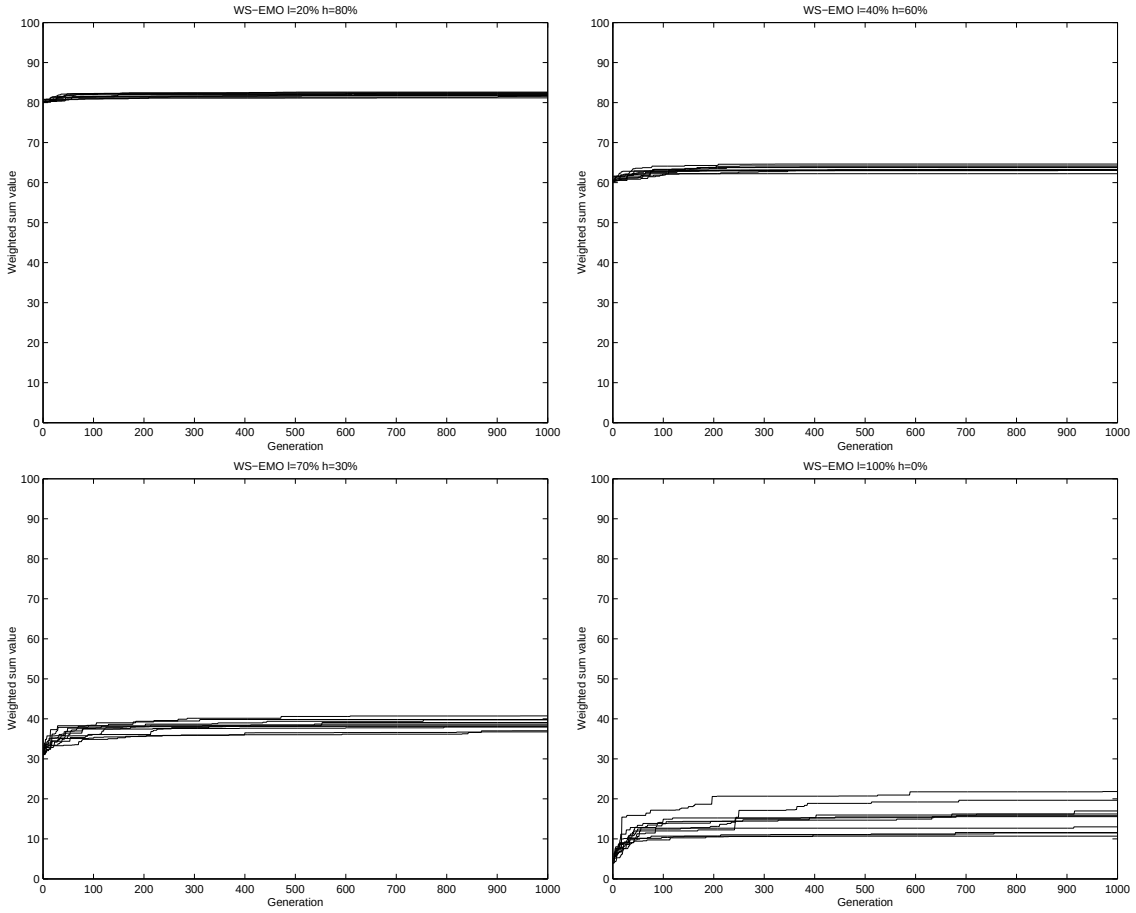


Figure 6.7: Best solutions obtained over 1000 generations for 10 runs using the weighted sum EMO algorithm with 1. $\gamma = 20\%$ (top left), 2. $\gamma = 40\%$ (top right), 3. $\gamma = 70\%$ (bottom left), 4. $\gamma = 100\%$ (bottom right). X-axis: Generation, Y-axis: Weighted sum value. Additional graphs can be found in the accompanying CD-ROM.

The evolution of the best solution using the weighted sum EMO algorithm for 10 runs over 1000 generations is shown in Figure 6.7 for four of the ten different combinations of weights assigned to the respective objectives. These graphs depict

the progression of the best solutions in terms of the actual weighted fitness value as evaluated using the weighted sum objective. The fixed weighting of the objectives appeared to have a significant impact on the improvement of the best solutions over time. When γ was set to low values as depicted in the top two graphs, the solutions were only able to improve over a highly constrained weighted value (Figures 6.7.1 & 6.7.2). This again was likely due to controllers using small numbers of hidden units being assigned high fitness values and hence dominating the elite solutions. As the value of γ was increased, the solutions were able to improve over a larger range of weighted values, although this also increased the variations between the best solutions found, as shown by the bottom two graphs (Figures 6.7.3 & 6.7.4). In the following paragraphs, we discuss the convergence of the solutions from the viewpoint of the separate component objectives.

The convergence of the best solution using the weighted sum EMO algorithm for 10 runs over 1000 generations is shown in Figure 6.8 in terms of the locomotion distance for the same four combinations of weights as in Figure 6.7. In the majority of the runs, most of the improvement achieved in terms of locomotion distance occurred very early during evolution, generally around the 100–120th generation. Also, the effect of rewarding controllers with smaller hidden layers can be seen clearly in Figure 6.8.3 where solutions with lower locomotion fitness but using less hidden units were accepted as the current best solution, causing this particular graph to have periods of apparently lower fitness during the convergence process. This in fact occurred because the weighted sum value of certain controllers that achieved greater locomotion distances were actually lower than controllers with less locomotion capabilities, as a result of using more hidden units in the hidden layer of the ANN compared to the less effective locomotion controllers. The progression of the use of hidden nodes over the evolutionary optimization process can be seen in the graphs that follow.

The convergence of the best solution using the weighted sum EMO algorithm for 10 runs over 1000 generations is shown in Figure 6.9 in terms of the number of hidden units used in the controller for the same four combinations of

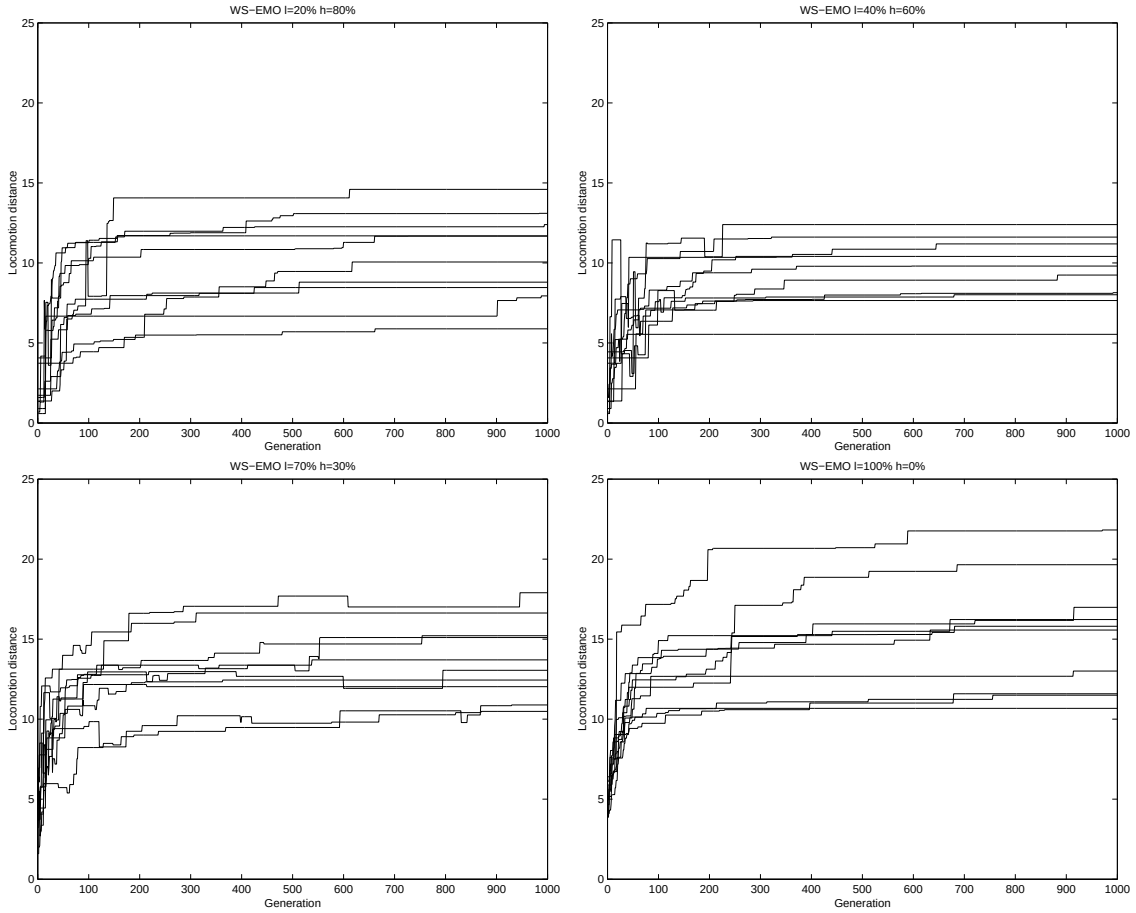


Figure 6.8: Locomotion distance of best solutions obtained over 1000 generations for 10 runs using the weighted sum EMO algorithm with 1. $\gamma = 20\%$ (top left), 2. $\gamma = 40\%$ (top right), 3. $\gamma = 70\%$ (bottom left), 4. $\gamma = 100\%$ (bottom right). X-axis: Generation, Y-axis: Locomotion distance. Additional graphs can be found in the accompanying CD-ROM.

weights as in the previous paragraphs. The effect of assigning the larger proportion of the weighted fitness to minimizing the hidden layer size can be seen clearly in the top two figures (Figures 6.9.1 & 6.9.2). The best solutions in these cases were highly constrained during the majority of the evolutionary process, using only 2 or less hidden nodes in the ANN controller. This consequently limited the ability of the solutions to improve on the locomotion distances achieved as a direct result of being able to only use controllers with very small numbers of hidden nodes. As γ is

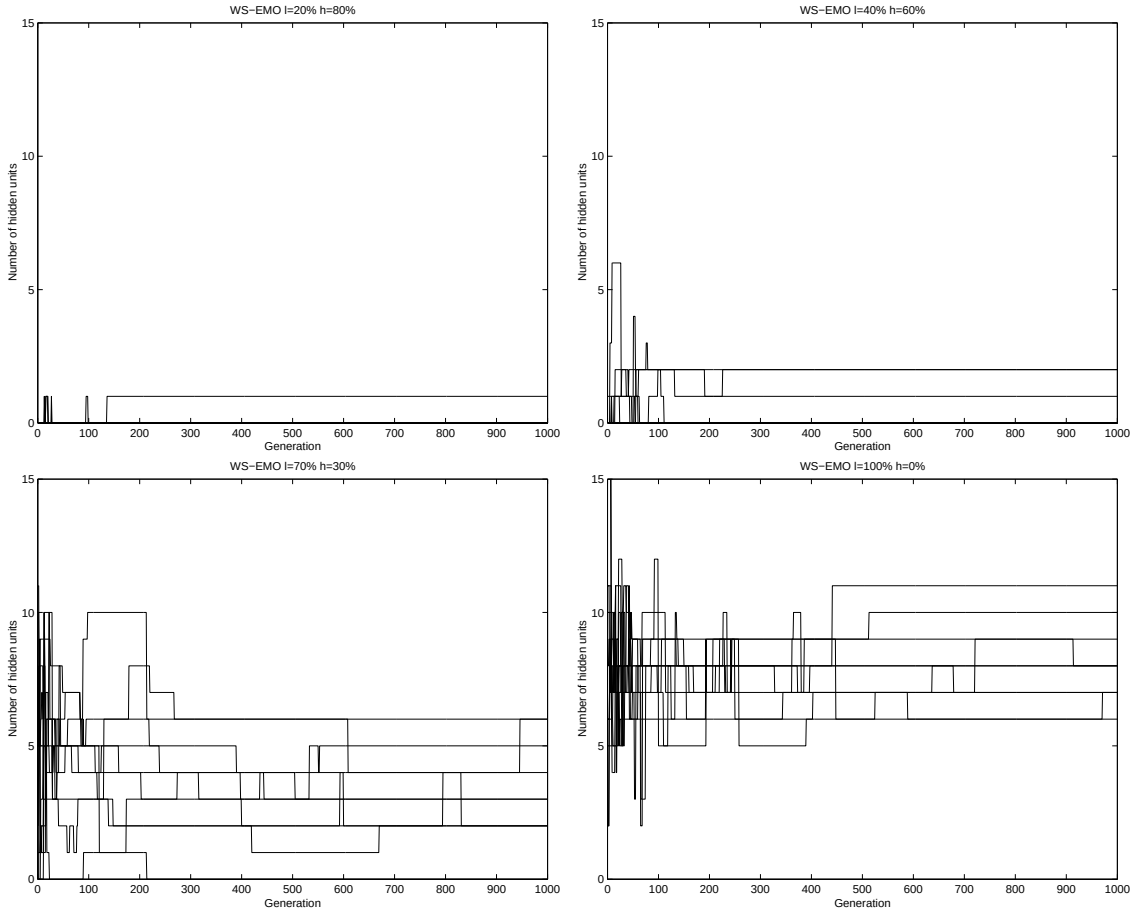


Figure 6.9: Hidden layer size of best solutions obtained over 1000 generations for 10 runs using the weighted sum EMO algorithm with 1. $\gamma = 20\%$ (top left), 2. $\gamma = 40\%$ (top right), 3. $\gamma = 70\%$ (bottom left), 4. $\gamma = 100\%$ (bottom right). X-axis: Generation, Y-axis: Number of hidden units. Additional graphs can be found in the accompanying CD-ROM.

increased to give more weight to the locomotion component, the constraint on the size of the hidden layer is lessened, thereby increasing the algorithm's likelihood of improving on the quality of the locomotion behavior by having more opportunities to experiment with ANN controllers with larger hidden layer sizes.

The mean locomotion distance and standard deviation for locomotion distance of the population as it evolved over 1000 generations using the weighted sum EMO algorithm are illustrated in Figures 6.10 and 6.11 respectively. Higher values

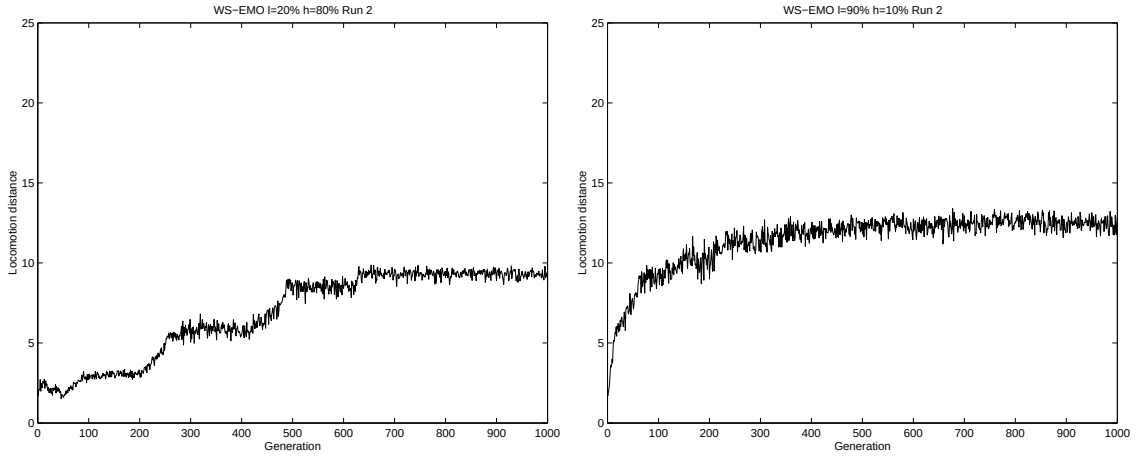


Figure 6.10: Mean locomotion distance of population over 1000 generations using the second seed for the weighted sum EMO algorithm with 1. $\gamma = 20\%$ (left), 2. $\gamma = 90\%$ (right). X-axis: Generation, Y-Axis: Locomotion distance. Additional graphs can be found in the accompanying CD-ROM.

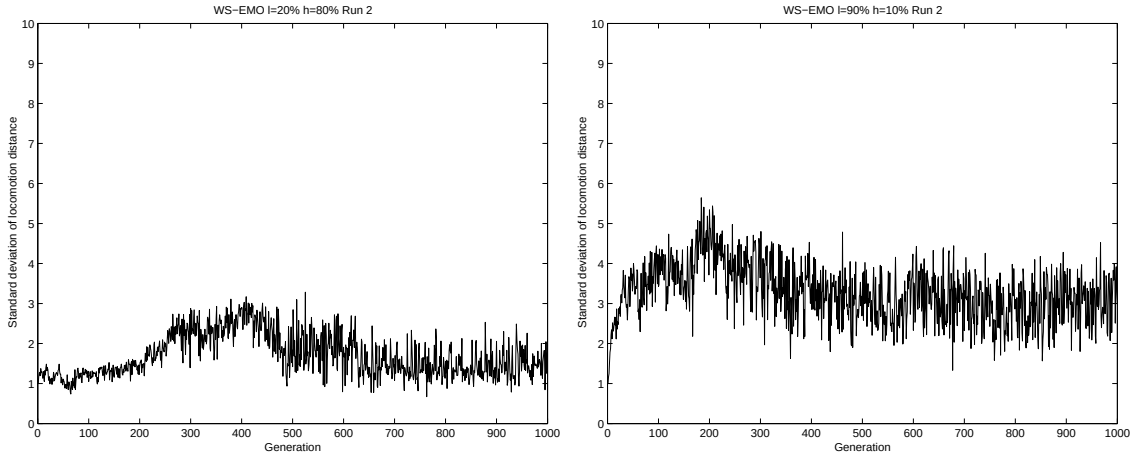


Figure 6.11: Standard deviation for locomotion distance of population over 1000 generations using the second seed for the weighted sum EMO algorithm with 1. $\gamma = 20\%$ (left), 2. $\gamma = 90\%$ (right). X-axis: Generation, Y-Axis: Standard deviation of locomotion distance. Additional graphs can be found in the accompanying CD-ROM.

of γ generally resulted in higher population means, which is expected as more weight is placed on optimizing the locomotion distance over minimizing the hidden layer.

A large majority of the population means reached a fairly constant range after the initial increase in fitness and either remained fixed within a small range or increased slightly in fitness as shown by Figure 6.10.2, which are similar to the trends observed in the hand-tuned EMO algorithm. Another interesting trend that emerged was that of the population mean increasing in a step-wise manner as depicted in Figure 6.10.1 but this only occurred in less than 10% of the runs. This phenomenon was most probably caused by the elite solutions becoming dominated by particular classes of solutions that changed their usage of the number of hidden units in the ANN controller and at the same time achieved distinctly better locomotion capabilities, resulting in short periods of constant fitness followed by significant jumps in fitness. The use of strong elitism in the weighted sum EMO algorithm also produced another common feature in almost all of the runs in that the population mean did not show any significant decrease in fitness over the evolutionary optimization process. This can be explained by the fact that the 15 best individuals representing the elite solutions carried forward from the previous generation will buffer any significant drop in the mean fitness of newly created individuals in the current generation. This observation is supported by the large movements of the standard deviation over time shown in Figure 6.11.2, which is representative of a large majority of the runs. These movements are noticeably larger than the standard deviations observed using the hand-tuned EMO algorithm where only the non-dominated solutions rather than elite solutions were retained. This strong elitism causes two distinct populations to emerge, one in the carried over individuals and another in the newly generated individuals. Consequently, when a number of new individuals are either good solutions similar to the elite solutions or bad solutions far removed from the elite solutions, the standard deviation will correspondingly change very significantly with strong elitism.

6.2.3 Search Space Characterization

The distribution of genotypes generated using the weighted sum EMO algorithm is plotted in Figure 6.12 in terms of locomotion distance and number of

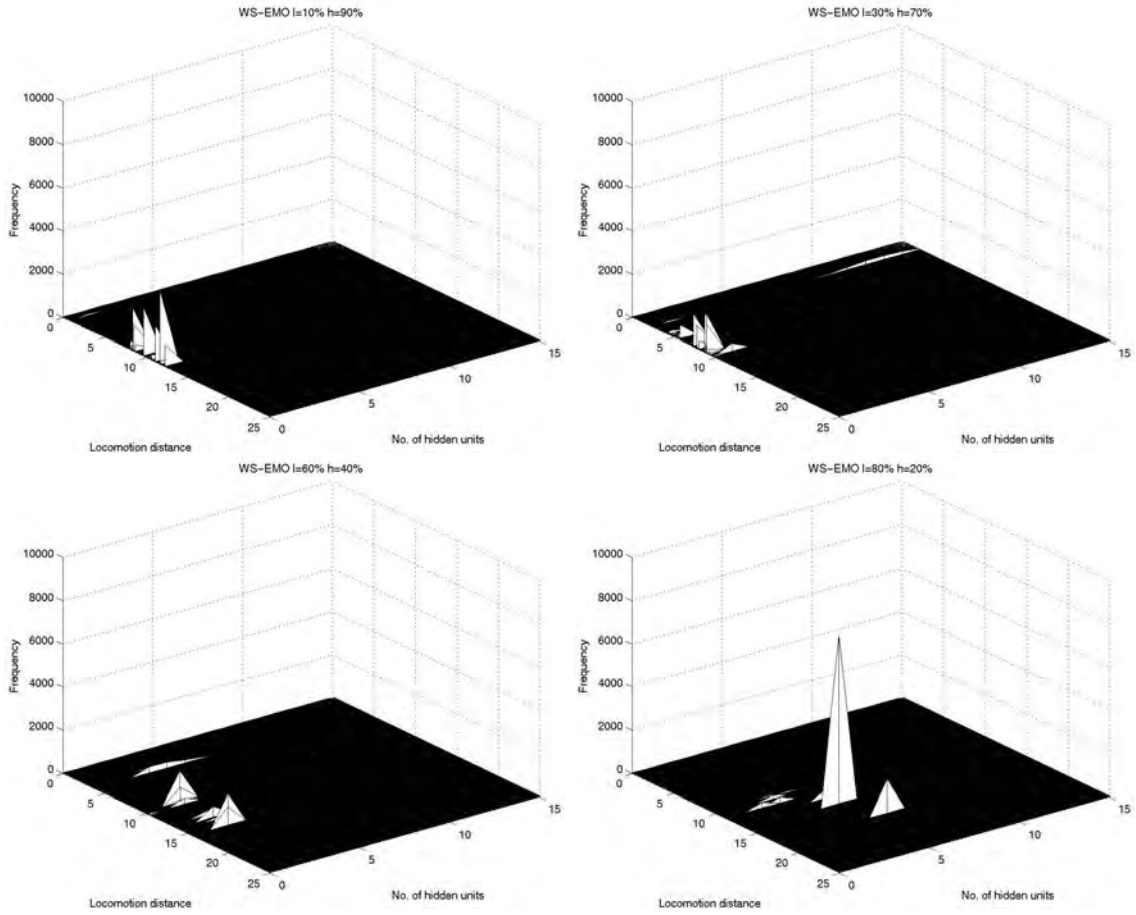


Figure 6.12: Frequency distribution of solutions obtained using the weighted sum EMO algorithm with 1. $\gamma = 10\%$ (top left), 2. $\gamma = 30\%$ (top right), 3. $\gamma = 60\%$ (bottom left), 4. $\gamma = 80\%$ (bottom right). X-axis: Locomotion distance, Y-axis: No. of hidden units, Z-axis: Frequency. Additional graphs can be found in the accompanying CD-ROM.

hidden units used in the ANN. Note that the frequency axis has been expanded from 140 in prior graphs to 10000 to cater for the higher concentrations of genotypes found within a specific range of objective values. Firstly, the distribution of genotypes across the objective spaces were dramatically different compared to SPANN and the hand-tuned EMO algorithm in that the generated genotypes were found to cluster very closely around highly specific values of locomotion distance and number of hidden units. The very significant change to the characteristics of the search

space is likely due to the use of a weighted sum approach coupled with an elitist approach. It is clear from the figures that genetic diversity in terms of the number of hidden units is not equivalent to that achieved with the Pareto approach. Allowing individuals to survive based solely on the weighted sum objective resulted in what can be seen in the figures, where hidden layers with certain numbers of hidden units dominated the evolutionary process. On the other hand, the carrying over of only non-dominated solutions in SPANN and the hand-tuned EMO algorithm leaves more room for variation since each parent is at least entirely different from the other in terms of the size of the hidden layer. Hence, the newly generated individuals can be expected to have greater diversity and consequently sample a larger proportion of the search space.

The contour graphs in Figure 6.13 illustrate the distribution of solutions across the two objectives of minimizing the hidden layer and maximizing locomotion distance. Two trends emerged when using the weighted sum approach in terms of the concentration of solutions across the respective objective spaces. The first trend is that of extremely high concentrations of solutions within very specific areas of the objective space, as evidenced by Figure 6.13.4. This phenomenon occurred when γ was set to 40%, 50%, 70% and 80%. The second group of genotypes had less highly concentrated distributions compared to the first group and had wider sampling of the search space as shown by Figures 6.13.1, 6.13.2 and 6.13.3, although this was still much less compared to the hand-tuned EMO algorithm and especially to SPANN. This supports the earlier observations that the respective weights assigned to the different objectives can significantly affect the behavior of the evolutionary optimization algorithm. As shown by this analysis, some combination of weights will cause the generated solutions to sample only a very limited area of the search space.

The probability density function of solutions obtained using the weighted sum EMO algorithm is illustrated in Figure 6.14. The probability of encountering different classes of solutions in terms locomotion capability varied considerably as different combinations of weights were used to evaluate the generated genotypes. As

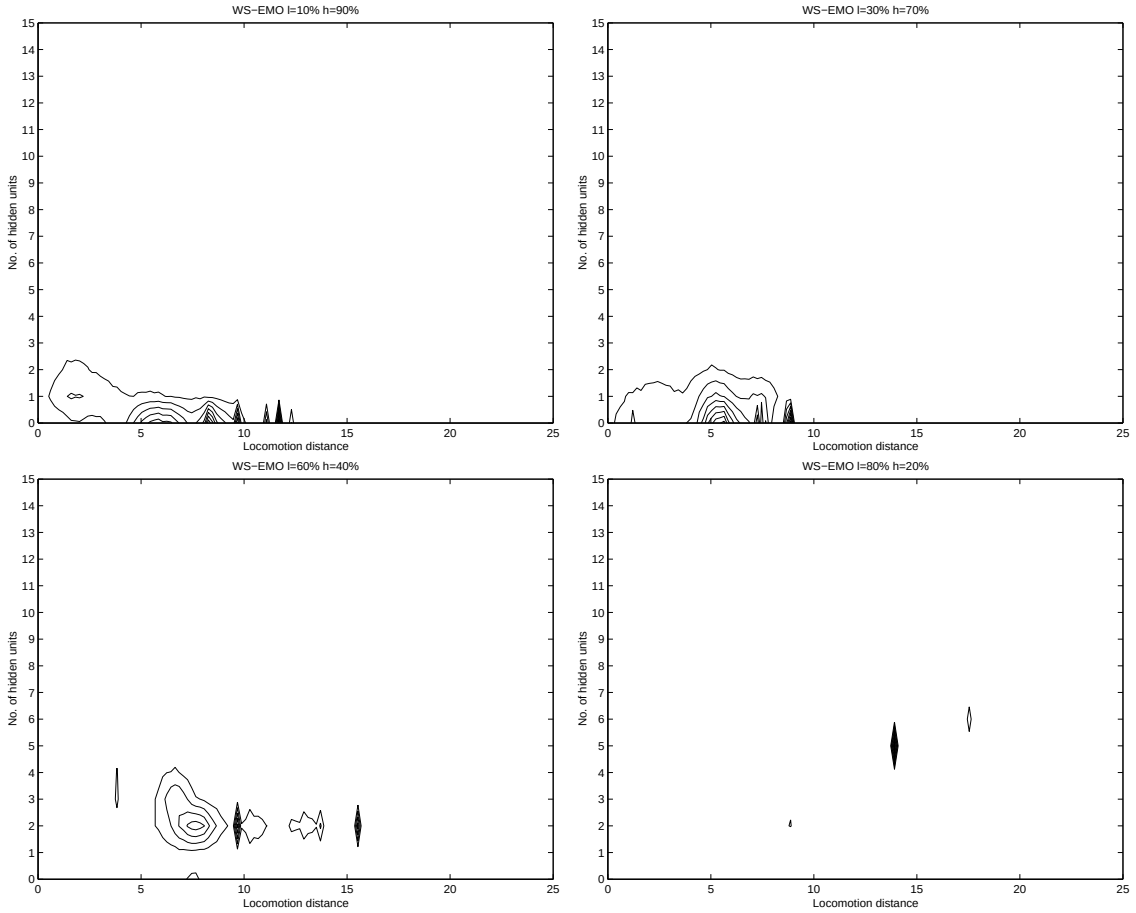


Figure 6.13: Contour graphs of frequency distribution of solutions obtained using the weighted sum EMO algorithm with 1. $\gamma = 10\%$ (top left), 2. $\gamma = 30\%$ (top right), 3. $\gamma = 60\%$ (bottom left), 4. $\gamma = 80\%$ (bottom right). X-axis: Locomotion distance, Y-axis: No. of hidden units. Additional graphs can be found in the accompanying CD-ROM.

expected, as more weight was placed on the locomotion component by increasing γ, the probability density curve could be seen to shift more to the right. However, as pointed out in earlier sections, certain combinations did not perform according to this expectation. Figure 6.14.2 shows that the probability of encountering solutions dropped to 0 as early as 9 units of locomotion distance. This rather poor performance is only slightly better than that achieved using random search (Figure 4.3.4), hill-climbing (Figure 4.7.4) and random walk (Figure 4.11.4) where the probability

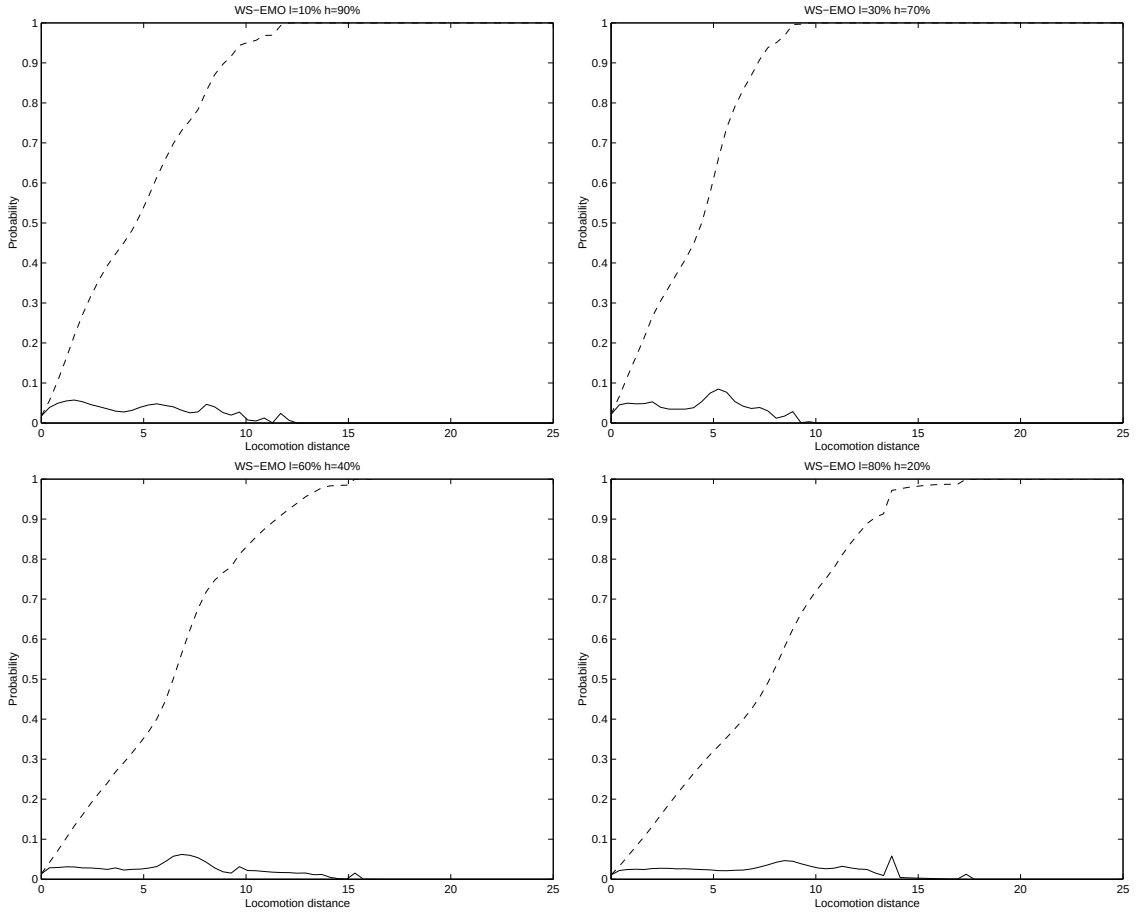


Figure 6.14: Density (solid) and cumulative (dashed) probability distribution of solutions obtained using the weighted sum EMO algorithm with 1. $\gamma = 10\%$ (top left), 2. $\gamma = 30\%$ (top right), 3. $\gamma = 60\%$ (bottom left), 4. $\gamma = 80\%$ (bottom right). X-axis: Locomotion distance, Y-axis: Probability. Additional graphs can be found in the accompanying CD-ROM.

dropped to 0 between 7 to 8 units of locomotion distance.

In summary, the search space characteristics of the EMO algorithm using a weighted sum approach pointed to the fact that although reasonably good sampling of the search space can be achieved with some weight combinations of the respective objectives, dramatically sparse sampling can similarly occur. Moreover, the sampling of the search space for the better combinations of weights were still significantly less uniformly distributed compared to SPANN and even the hand-tuned

EMO algorithm to a lesser extent.

6.3 A Single-Objective EA

6.3.1 Experimental Setup

In the last set of experiments, we used a conventional EA which optimizes only one objective as opposed to optimization of multiple objectives in EMO algorithms such as SPANN. The only objective being optimized in the following evolutionary runs is the locomotion distance achieved by the creature's ANN controller while the size of the hidden layer is kept fixed. Hence, only the f_1 fitness function is used to evaluate the genotypes. Apart from the change of optimizing two objectives to one, the single-objective algorithm is otherwise similar to the SPANN algorithm in all other respects. The crossover and mutation rates are self-adaptive and are identical to their counterparts in SPANN except that crossover and mutation now excludes any changes to the number of hidden units used in the ANN controller since this component is fixed in the single-objective EA. As in previous comparisons, the NNType3 architecture was used in this set of experiments and all other parameters remained the same: 1000 generations, 30 individuals, 500 timesteps and 10 repeated runs. As in the weighted sum EMO algorithm discussed in Section 6.2, the $(\mu + \lambda)$ strategy is used in this single-objective EA where the 15 best individuals of the population are carried over to the next generation without any modification to the genotype at all. Sixteen separate sets of evolutionary runs were conducted corresponding to each one of the different number of nodes used in the hidden layer ranging from 0 to 15, which is the range allowed in the multi-objective runs. As with prior algorithms used in this chapter, the individual genotypes generated were grouped into 6250 discrete intervals for search space characterization.

6.3.2 Results and Discussion

The results obtained from using the single-objective EA for conducting the EMO process are given in Table 6.3. The highest overall best f_1 fitness of 22.4 was

No. of Hidden Units	Overall Best Fitness	Worst of the Best Fitness	Average Best Fitness ± Standard Deviation
0	20.3725	11.6791	15.7516 ± 2.9721
1	19.3005	12.2596	15.1441 ± 2.0260
2	19.8772	9.9934	16.3236 ± 2.7242
3	19.2861	9.5247	15.1532 ± 3.1696
4	20.6868	12.9391	15.2088 ± 2.2106
5	19.9139	11.2756	15.1562 ± 2.8741
6	19.6655	12.7716	16.0317 ± 2.0719
7	17.8093	13.4571	15.8033 ± 1.6159
8	21.6668	12.1226	17.4358 ± 3.2508
9	20.4605	12.1012	15.7375 ± 2.6430
10	19.4172	13.5765	16.1514 ± 2.1318
11	21.6224	9.3723	15.0614 ± 3.5612
12	22.3296	11.6783	15.4287 ± 3.0020
13	17.6432	12.1548	15.0359 ± 1.8909
14	22.4069	10.9295	16.6273 ± 2.8095
15	19.7747	12.5919	15.6150 ± 2.4605

Table 6.3: Best solutions obtained over 10 independent runs using the single-objective EA with different hidden layer sizes.

obtained using a hidden layer of 14 nodes while the lowest overall best fitness of 17.6 was obtained using a hidden layer of 13 nodes. In terms of average best locomotion distance achieved, the setup in which the ANN controller's hidden layer was fixed to use 8 nodes provided the best result while the worst result was obtained when the hidden layer was fixed at 11 nodes. In general, the variations between the best solutions achieved were high when using the single-objective EA. Only 2 out of the 16 different hidden layer setups had standard deviations of less than 2, 10 setups had standard deviations of between 2 and 3 while the remaining 4 setups had standard deviations of more than 3 units distance. The setup which used 11 hidden units also had the highest standard deviation where the difference between the best and worst solutions obtained was 12.2, which is a variation of more than 56% of the overall best fitness achieved using this hidden layer setup. These observations suggest that optimizing the artificial creature's controllers using the single-objective EA is quite unstable and the quality of solutions obtained can vary greatly between different runs.

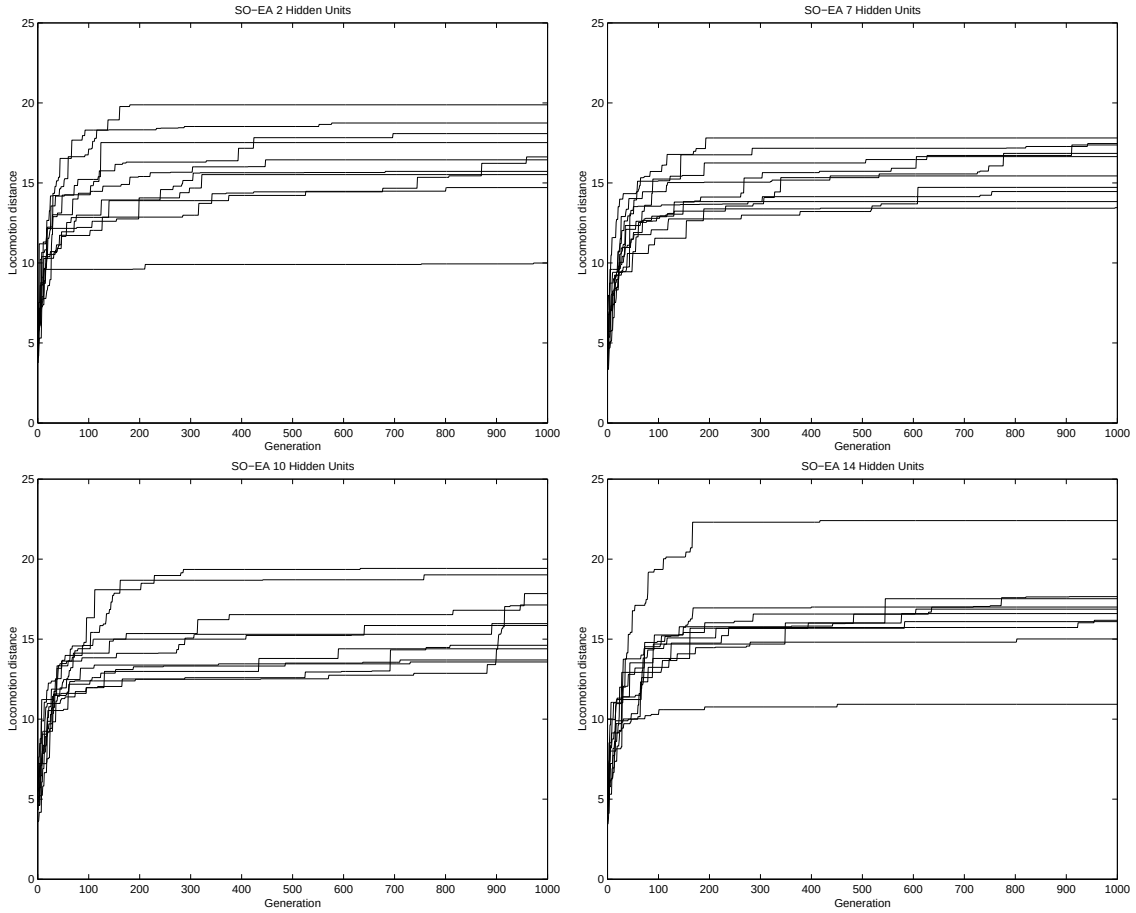


Figure 6.15: Best fitness for solutions obtained over 1000 generations for 10 runs using the single-objective EA with 1. 2 hidden units (top left), 2. 7 hidden units (top right), 3. 10 hidden units (bottom left), 4. 14 hidden units (bottom right). X-axis: Generation, Y-axis: Locomotion distance. Additional graphs can be found in the accompanying CD-ROM.

The evolution of the best solution using the single-objective EA for 10 runs over 1000 generations is shown in Figure 6.15 for four of the sixteen different hidden layer setups. Across all the different sizes of hidden layer used in the ANN controller, the majority of the improvement achieved in the best solutions occurred very early during evolution, as in the weighted sum method. This phenomenon could be seen to occur as early as the 50–80th generation in some of these runs. This can be explained by the fact that all the optimization effort is being focused solely on

a single objective and thus the single-objective EA should show a faster rate of improvement and subsequently be able to converge earlier compared to algorithms with distinct multiple objectives. However, the cost of this type of fast convergence is as discussed earlier in the previous paragraph, that the standard deviations between the best evolved solutions can be quite large. Figure 6.15.4 depicts this phenomenon clearly and is representative of the different hidden layer sizes used in the other runs for this single-objective EA. This shows that although very good solutions can be obtained in terms of locomotion distance, correspondingly poor solutions can also be expected.

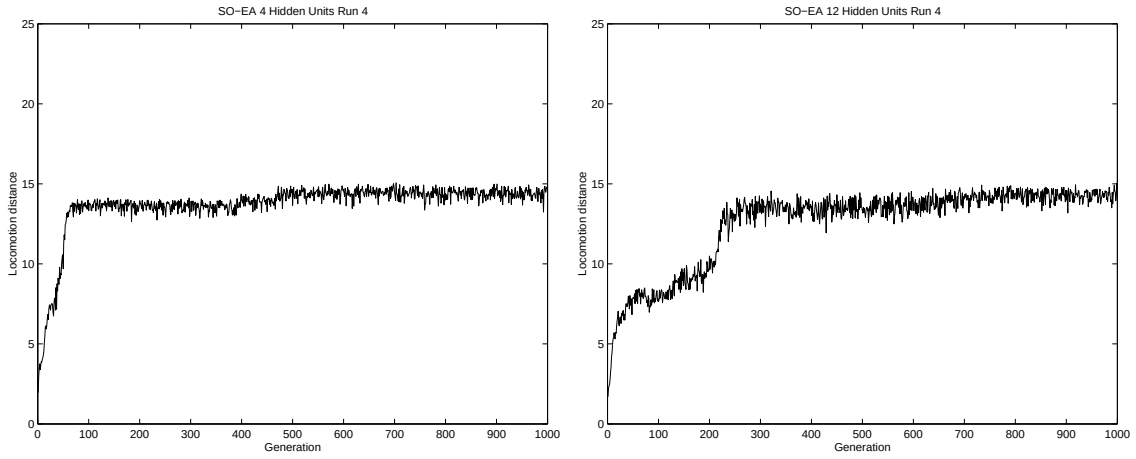


Figure 6.16: Mean fitness of population over 1000 generations using the fourth seed for the single-objective EA algorithm with 1. 4 hidden units (left), 2. 12 hidden units (right). X-axis: Generation, Y-Axis: Locomotion distance. Additional graphs can be found in the accompanying CD-ROM.

The mean locomotion distance and standard deviation for locomotion distance of the population as it evolved over 1000 generations using the single-objective EA is illustrated in Figures 6.16 and 6.17 respectively. The movement of the population means and standard deviations were very similar to the trends observed in the weighted sum EMO algorithm. The mean could be seen to increase and then either remain fixed or increase slightly over time (Figures 6.16.1 & 6.16.2). The mean also did not show any significant decrease in fitness throughout evolution.

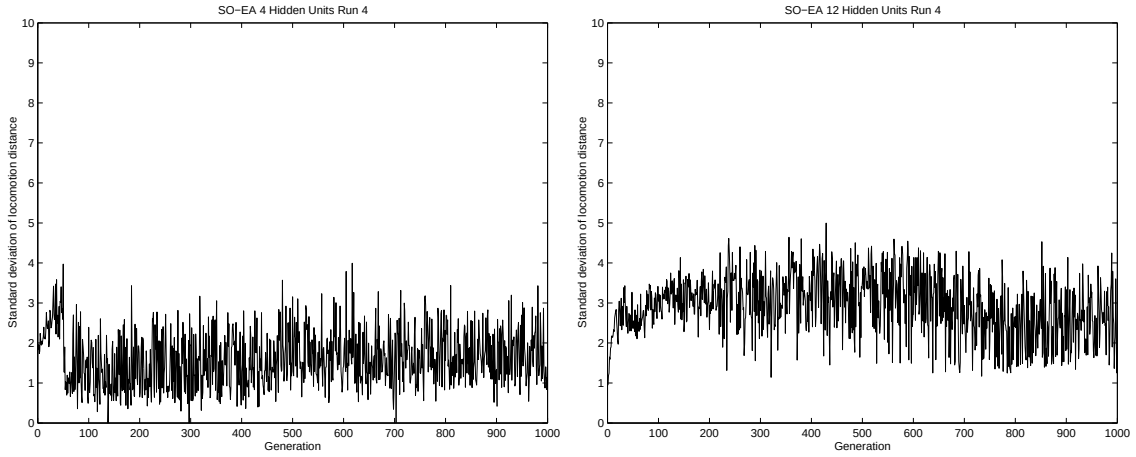


Figure 6.17: Standard deviation for fitness of population over 1000 generations using the fourth seed for the single-objective EA algorithm with 1. 4 hidden units (left), 2. 12 hidden units (right). X-axis: Generation, Y-Axis: Standard deviation of locomotion distance. Additional graphs can be found in the accompanying CD-ROM.

Likewise, the standard deviation movements (Figures 6.17.1 & 6.17.2) were again noticeably larger than those observed using the hand-tuned EMO algorithm. The highly similar trends observed in this single-objective EA and the weighted sum EMO algorithm strongly suggest that the use of strong elitism in both these algorithms is the likely cause of these phenomena. The effect of different sizes of hidden layers used in the controller did not appear to create any consistently different trends in the behavior of the population mean over time. However, the standard deviation of the population appeared to have varied within a lower range of deviation for the controllers that used less hidden units compared to those that used more hidden units. This observation suggests that although the population means were on average quite similar across different sizes of hidden layer used, the discrepancy between the best solutions and the newly generated solutions were larger in controllers that used more hidden units. Therefore, the analysis of the populations' average fitness and standard deviations points to the fact that the fitness landscapes may be more rugged for larger hidden layer sizes since newly produced offspring are further away

from the elite parents compared to controllers with smaller hidden layer sizes.

6.3.3 Search Space Characterization

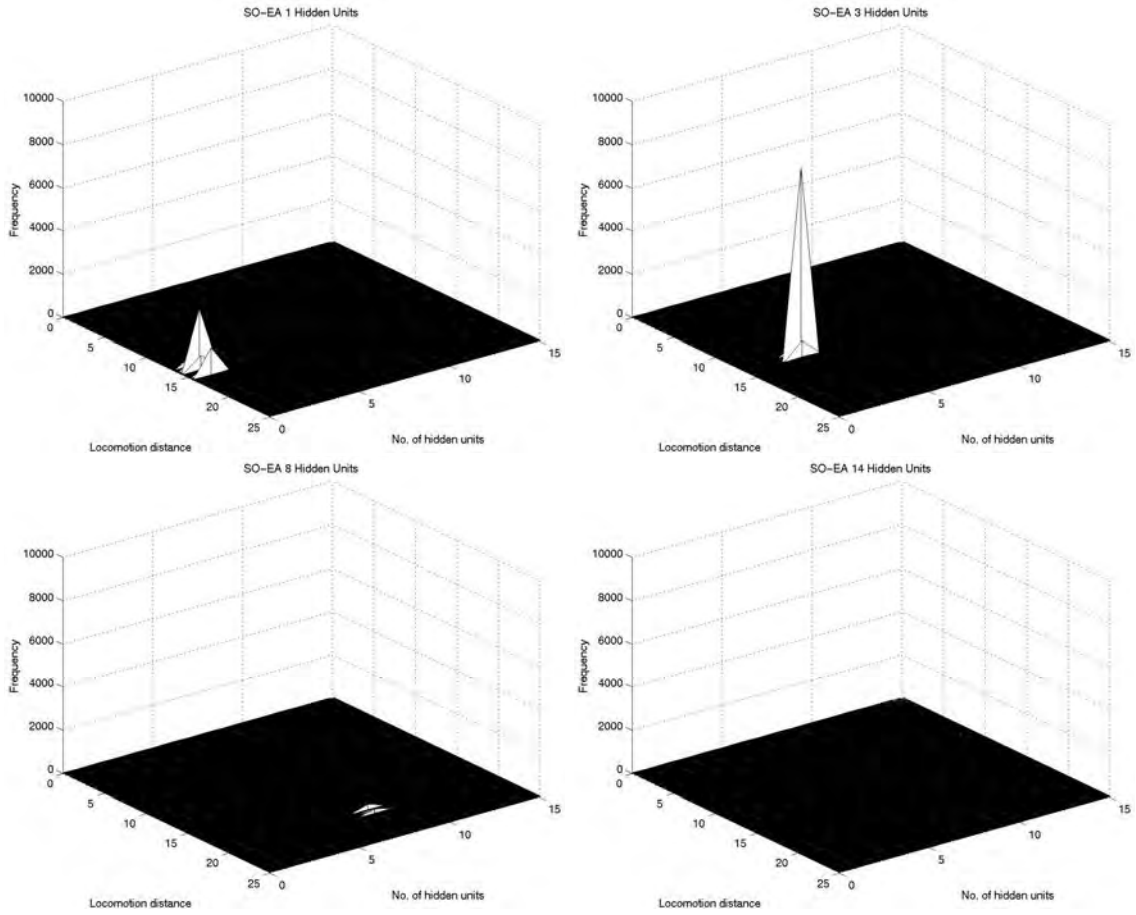


Figure 6.18: Frequency distribution of solutions obtained using the single-objective EA with 1. 1 hidden unit (top left), 2. 3 hidden units (top right), 3. 8 hidden units (bottom left), 4. 14 hidden units (bottom right). X-axis: Locomotion distance, Y-axis: No. of hidden units, Z-axis: Frequency. Additional graphs can be found in the accompanying CD-ROM.

The distribution of genotypes generated using the single-objective EA is plotted in Figure 6.18 in terms of locomotion distance and number of hidden units used in the ANN. Note that the frequency axis was again expanded to 10000 as in the analysis of the weighted sum approach (Section 6.2.3) to cater for the higher con-

centrations of solutions. The characteristics of the genotype distribution in the objectives spaces were expectedly very different from all prior algorithms since the size of the hidden layer was forcibly maintained within each evolutionary optimization setup by virtue of the single-objective methodology. Two different trends emerged from these runs. The distribution of solutions was more highly clustered when smaller hidden layer sizes were used, especially in setups that used between 0 and 4 hidden units. The distribution of genotypes depicted in Figures 6.18.1 and 6.18.2 are representative of this first trend. As the size of the hidden layer was increased, the generated genotypes were less closely concentrated, large peaks in the frequency were less commonly seen, and the magnitude of areas with high concentrations was also lower. The distribution of genotypes depicted in Figures 6.18.3 and 6.18.4 are representative of this second trend. Frequency distribution graphs at higher resolutions are shown below for the latter two hidden layer sizes.

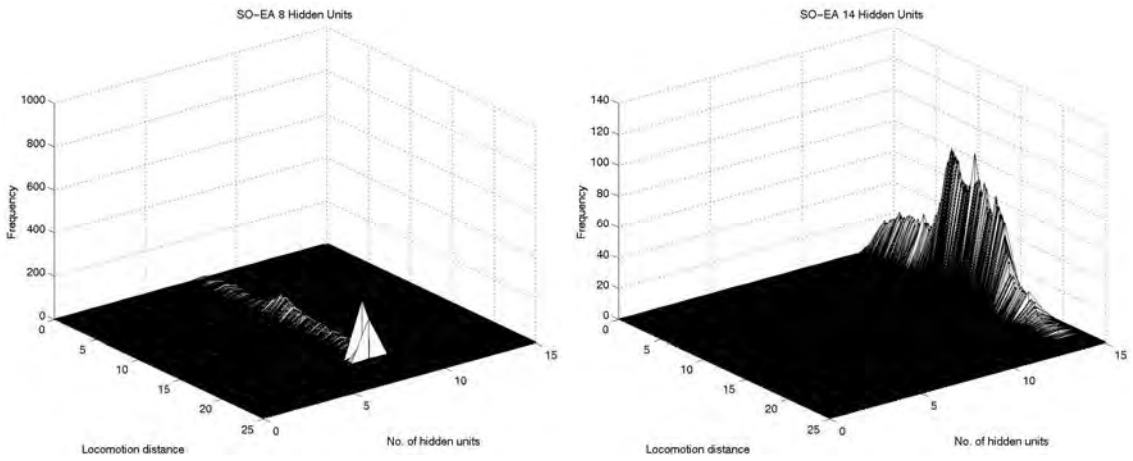


Figure 6.19: Frequency distribution of solutions obtained using the single-objective EA at higher resolutions for 1. 8 hidden units with frequency axis re-scaled to 1000 (left), 2. 14 hidden units with frequency axis re-scaled to 140 (right). X-axis: Locomotion distance, Y-axis: No. of hidden units, Z-axis: Frequency. Additional graphs can be found in the accompanying CD-ROM.

The distribution of controllers generated using 8 and 14 hidden units are illustrated in Figure 6.19 at increased resolutions to show the finer characteristics

of these fitness landscapes. Figure 6.19.1 is re-scaled to a frequency of 1000 and Figure 6.19.2 is re-scaled to a frequency of 140. Some clustering of genotypes can still be observed in controllers that used 8 hidden units. However, in controllers that used 14 hidden units, the distribution of genotypes was more uniformly distributed across the objective space. This phenomenon can be further explained by analyzing the associated contour graphs of these distributions.

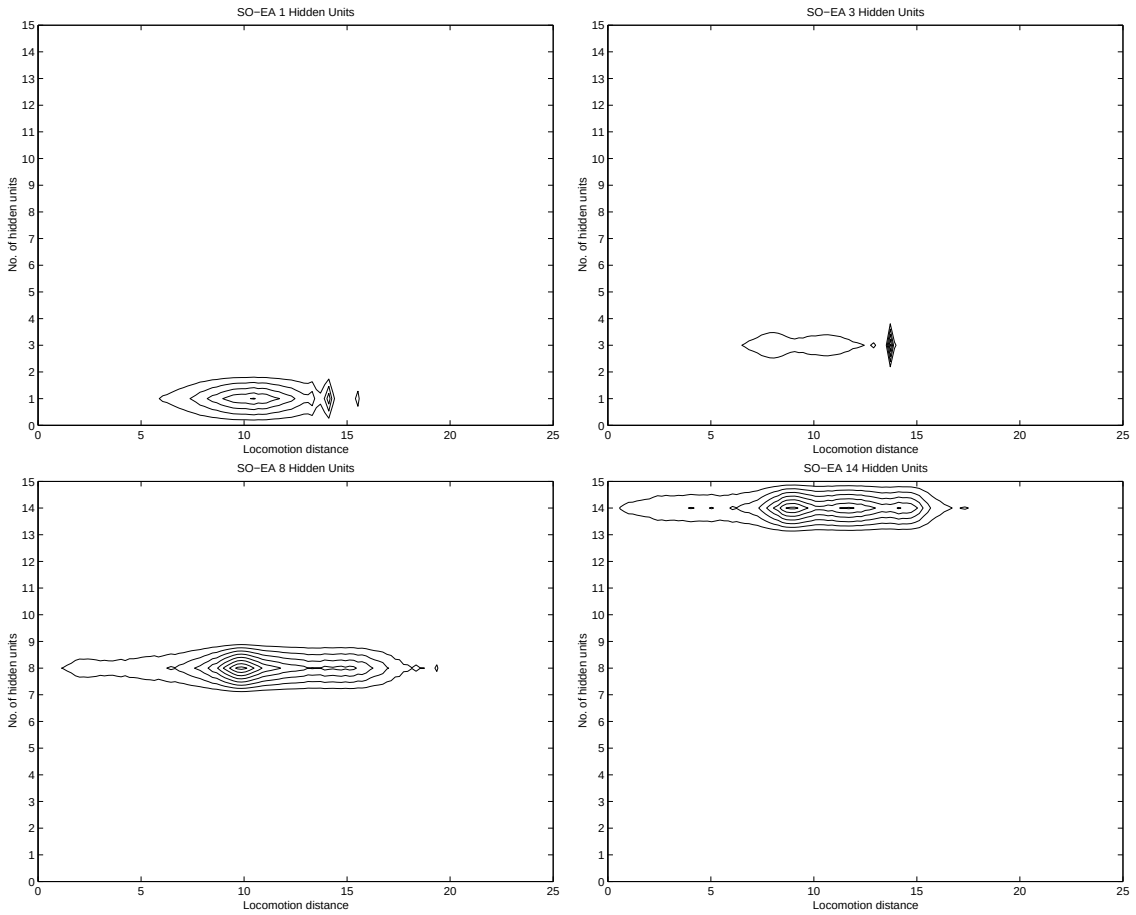


Figure 6.20: Contour graphs of frequency distribution of solutions obtained using the single-objective EA with 1. 1 hidden unit (top left), 2. 3 hidden units (top right), 3. 8 hidden units (bottom left), 4. 14 hidden units (bottom right). X-axis: Locomotion distance, Y-axis: No. of hidden units. Additional graphs can be found in the accompanying CD-ROM.

The contour graphs in Figure 6.20 illustrate the distribution of solutions

across the two objectives of minimizing hidden layer size and maximizing locomotion distance. The two trends that emerged in the 3D graphs discussed in the previous paragraph can be seen again clearly in these contour graphs. With a smaller number of hidden units, the solutions were distributed within a much smaller range of locomotion values, where most controllers achieved locomotion distances of between 6 and 14 units (Figures 6.20.1 & 6.20.2). Using a larger number of hidden units, the genotypes generated were able to sample larger areas of the objective space, where a significant proportion of the controllers were able to achieve locomotion distances that ranged between 1 and 17 units (Figures 6.20.3 & 6.20.4). This suggests that as the size of the hidden layer increases, the artificial creature is able to generate a wider range of locomotion capabilities in terms of the overall distance moved as a result of being able to sample the much larger search space offered by the increased network sizes.

The probability density function of solutions obtained using the single-objective EA is illustrated in Figure 6.21. Again, these graphs show that the smaller hidden layer sizes produced lower quality controllers in terms of locomotion distance compared to larger hidden layer sizes. Four hidden layer sizes of 8, 11, 12 and 14 were able to sample controllers up to 19 units of distance in the class of larger-sized networks before the probability dropped to 0. The graphs depicted in Figures 6.21.3 and 6.21.4 are indicative of the probability density functions obtained using these setups with larger hidden layer sizes. In comparison, for seven out of the eight hidden layer sizes that used 7 nodes or less, the probability of obtaining controllers dropped to 0 only around distances of between 15 and 17 units, as shown by the cumulative curves. The graphs depicted in Figures 6.21.1 and 6.21.2 are indicative of the probability density functions obtained using these setups with smaller hidden layer sizes.

In general, the search space characteristics were indicative of the fact that genotypes generated using smaller-sized hidden layers produced locomotion capabilities that were more constrained in terms of the range of distances achieved by the creature compared to genotypes that used larger-sized hidden layers. In the latter

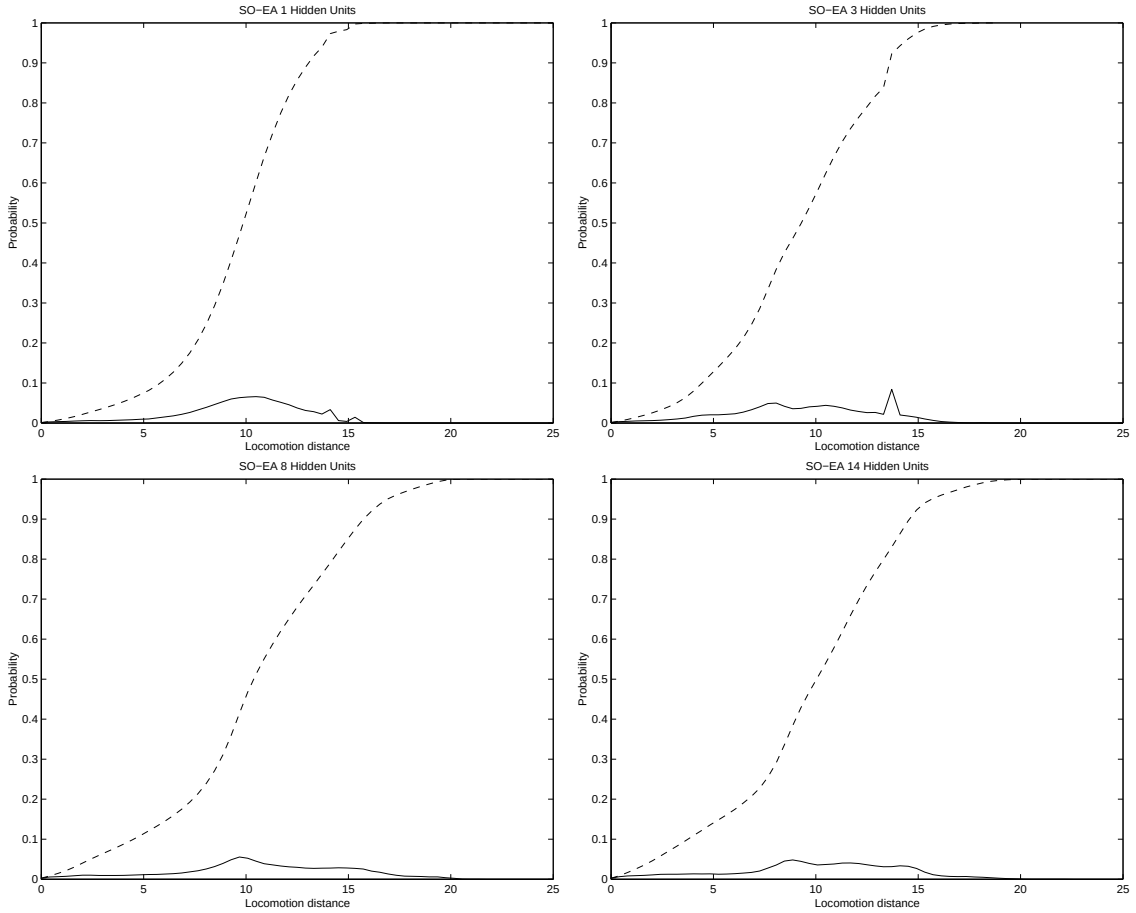


Figure 6.21: Density (solid) and cumulative (dashed) probability distribution of solutions obtained using the single-objective EA with 1. 1 hidden unit (top left), 2. 3 hidden units (top right), 3. 8 hidden units (bottom left), 4. 14 hidden units (bottom right). X-axis: Locomotion distance, Y-axis: Probability. Additional graphs can be found in the accompanying CD-ROM.

group of controllers, a significantly larger proportion of the newly generated genotypes were sampled from higher fitness sub-spaces. However, since there were no observable trends in terms of the best controllers found (see Table 6.3), these results strongly suggest that the underlying fitness landscape becomes increasingly rugged as the hidden layer size increases and coupled with the much larger search spaces, much more optimization effort may be required to find increasingly fitter solutions in these larger networks. This is an expected outcome given that the VC-dimension

increases with an increase in the number of hidden units (as explained earlier in Section 1.3) — therefore, more optimization effort is required.

6.4 Comparing SPANN Against Conventional Evolutionary Optimization Approaches

In this section, we compare the results obtained using the three EAs presented in this chapter against the results obtained using the SPANN algorithm. In the first section, SPANN is compared against the hand-tuned method, followed by a comparison against the weighted sum method in the second section, and then against the single-objective method in the third section. In the next two sections, firstly an overall discussion of the global Pareto solutions using SPANN as well as all other algorithms employed thus far is given in terms of the trade-off between the quality of the locomotion controller generated against the computational cost involved in obtaining these controllers, which is followed by an investigation into the amount of redundancy that is present in the overall best controllers evolved for locomotion distance using SPANN against the hand-tuned, weighted sum and single-objective methodologies. Finally, SPANN is compared to a well-known, state-of-the-art Pareto EMO algorithm called the Non-dominated Sorting Genetic Algorithm II (NSGA-II) (Deb, Agrawal, Pratab, and Meyarivan 2000).

6.4.1 SPANN Against a Hand-Tuned EMO Algorithm

The advantage of using the self-adaptive Pareto approach against hand-tuning of crossover and mutation rates of the EMO algorithm is that it reduces the number of repeated experiments required to find the "right" combination of these parameters in order to generate the best possible solutions. Furthermore, it is unclear what effects fixing the crossover and mutation rates throughout the evolutionary optimization process will have on the quality of the eventual solutions obtained. In the results below, we compare the best solutions obtained from using 9 different combinations of crossover and mutation rates using the hand-tuned version

of the EMO algorithm against the best solutions obtained using the self-adaptive SPANN EMO algorithm.

Algorithm	Average Best Locomotion Distance ± Standard Deviation	t-statistic (against SPANN)	No. of Hidden Units
SPANN	13.9626 ± 1.7033	-	4.9 ± 2.6
HT-EMO **c**=10% **m**=10%	13.5192 ± 2.7845	(0.48)	3.2 ± 1.8
HT-EMO **c**=10% **m**=50%	14.1158 ± 2.6535	0.16	6.6 ± 2.1
HT-EMO **c**=10% **m**=90%	13.2843 ± 1.8225	(0.94)	8.1 ± 2.7
HT-EMO **c**=50% **m**=10%	14.1268 ± 2.1286	0.16	3.0 ± 2.4
HT-EMO **c**=50% **m**=50%	15.3819 ± 2.3195	1.39	6.5 ± 1.7
HT-EMO **c**=50% **m**=90%	13.1881 ± 1.4715	(1.03)	7.7 ± 2.5
HT-EMO **c**=90% **m**=10%	14.1511 ± 2.4721	0.20	3.4 ± 1.8
HT-EMO **c**=90% **m**=50%	13.1978 ± 1.6447	(1.42)	6.1 ± 1.4
HT-EMO **c**=90% **m**=90%	12.1653 ± 2.3799	(1.61)	6.8 ± 2.9

Table 6.4: Comparison of best locomotion distance for Pareto solutions obtained over 10 independent runs using the SPANN and hand-tuned EMO (HT-EMO) algorithms. **c** = crossover rate, **m** = mutation rate.

As shown in Table 6.4, the use of hand-tuned crossover and mutation rates did not provide any significant advantage over the SPANN algorithm in terms of the average best locomotion distance achieved by the evolved controllers. A t-test showed no significant differences at the $\alpha = 0.05$ and $\alpha = 0.01$ significance levels. Four combinations of the hand-tuned EMO algorithm gave marginally better results over the 10 runs in terms of the average best solution obtained while five other combinations performed worse than the SPANN algorithm. In terms of the number of hidden units used, three combinations in the hand-tuned EMO algorithm used an average of 1.7 nodes less than SPANN while six other combinations used an average of 7.0 nodes more than the SPANN algorithm. As such, the self-adaptive SPANN algorithm is beneficial compared to a hand-tuned EMO algorithm in that it reduces the computational runs required while still being able to maintain the same quality of solutions generated.

6.4.2 SPANN Against a Weighted Sum EMO Algorithm

A weighted sum approach combines the two objectives into a single objective by taking a weighted sum of the objectives. There are four main advantages of using the Pareto approach over a weighted sum method:

- The weighted sum method would only be able to generate a single Pareto solution in a single run compared to an entire set of Pareto solutions in a single run using the Pareto approach. Multiple runs will be required to generate a Pareto-front when using the weighted sum method.

- The determination of the weights is arbitrary in a weighted sum method. Some form of hand-tuning these weights will need to be carried out in order to obtain good results and as such, extra runs will again be required compared to the Pareto approach.

- The different objectives combined in a weighted sum method are assumed to be somehow commensurable, that is the objectives can be measured in the same units. In the case where they are not, as in this case of combining locomotion distance and number of hidden units, the use of the correct relative weights will be necessary to overcome this problem. Again, the Pareto approach does not require any such assumption to hold true since it treats each objective independently from the other.

- The weighted sum method assumes that the Pareto-front of the multi-objective optimization problem is of a convex nature. If the Pareto-front of the multi-objective optimization problem is actually non-convex, then the Pareto solutions generated by the weighted sum method will result in a discontinuous Pareto-front since the single-objective hyperplane will not be able to sample the non-convex regions of the Pareto-front. As such, in order to use a weighted sum method, the experimenter will first need to ascertain whether the particular problem is convex or otherwise, and no such information is usually available until the actual experiments are carried out and the Pareto-front plotted. Conversely, knowledge of such properties about the multi-objective optimization

problem is not required since the solutions generated using a Pareto approach is not constrained or limited to a specific Pareto-front of the multi-objective optimization problem.

Algorithm	Average Best Locomotion Distance ± Standard Deviation	t-statistic (against SPANN)	No. of Hidden Units
SPANN	13.9626 ± 1.7033	-	4.9 ± 2.6
WS-EMO γ=10%	9.8571 ± 1.4277	(5.97)	0.0 ± 0.0
WS-EMO γ=20%	10.4613 ± 2.6883	(3.19)	0.1 ± 0.3
WS-EMO γ=30%	8.4306 ± 1.3288	(7.96)	0.1 ± 0.3
WS-EMO γ=40%	9.4011 ± 2.1017	(4.46)	0.5 ± 0.8
WS-EMO γ=50%	11.3924 ± 3.0330	(2.15)	0.8 ± 0.9
WS-EMO γ=60%	12.1794 ± 2.9865	(1.86)	1.6 ± 1.0
WS-EMO γ=70%	13.7448 ± 2.4376	(0.33)	3.3 ± 1.9
WS-EMO γ=80%	14.0521 ± 2.3034	0.09	4.1 ± 1.5
WS-EMO γ=90%	15.1119 ± 1.9977	1.76	5.6 ± 1.9
WS-EMO γ=100%	15.2829 ± 3.6578	1.04	8.1 ± 1.5

Table 6.5: Comparison of best locomotion distance for Pareto/best solutions obtained over 10 independent runs using the SPANN and weighted sum EMO (WS-EMO) algorithms. γ = relative weight parameter.

In Table 6.5, we compare the weighted sum EMO against the Pareto SPANN algorithm. Results comparable to those obtained using the SPANN algorithm are achieved only with $\gamma = 70\%$. Although slightly higher locomotion distances were achieved using higher values of γ, which places more emphasis on the locomotion component of the weighted objective function, in all cases the standard deviation of the solutions were higher for the average best fitness for locomotion distance. Also, the case where $\gamma = 100\%$, which does not put any pressure whatsoever towards optimizing the size of the hidden layer, results in a very high average of hidden units used in the evolved controllers. This suggests that a significant amount of redundancy may be present in these networks, given that a t-test showed none of these weighted sum solutions were significantly better than those obtained with SPANN at both the $\alpha = 0.05$ and $\alpha = 0.01$ significance levels. Conversely, three of the weight combinations resulted in solutions significantly worse than SPANN at

the $\alpha = 0.01$ significance level ($\gamma = 10\%, 30\%, 40\%$) and one weight combination worse than SPANN at the $\alpha = 0.05$ significance level ($\gamma = 20\%$). Therefore, obtaining good solutions when using a weighted sum method critically depends on the choice of weights used on the respective objective functions and to find this right combination of weights would require multiple evolutionary runs to be conducted. Hence, the Pareto approach adopted in our SPANN algorithm is preferable from a computational cost point of view over a weighted sum method since it is able to proceed with the evolutionary optimization process without any tuning of weights and is still able to produce highly competitive results.

6.4.3 SPANN Against a Single-Objective EA

In a single-objective EA, the number of objectives that can be optimized in any one run is restricted to one. If there is more than one factor that may affect the quality of the solutions that are obtained from the evolutionary optimization process, then the single-objective EA will need to be re-run multiple times to test the effects of these other factors. For example, to test the effect of the size of the number of hidden units used on the evolution of artificial creature controllers, a separate set of runs will need to be carried out for each hidden layer size (see the experiments reported in Bongard and Pfeifer (2002) for such a case). As such, the obvious advantage an EMO algorithm has over a single-objective EA is its ability to optimize multiple objectives simultaneously, thereby significantly reducing the number of computational runs required to investigate other factors that may be crucial to the effectiveness of the evolutionary process.

In Table 6.6, we compare the single-objective EA against the multi-objective SPANN algorithm. In all the single-objective runs, higher locomotion distances were achieved by the evolved controllers in terms of the mean of the best solutions compared to SPANN. This is expected since all the evolutionary optimization pressure is focused only on the one objective of maximizing locomotion distance whereas this pressure is halved in the EMO case, where it is being shared with the objective of minimizing the hidden layer size. However, none of these results were significantly

Algorithm	No. of Hidden Units	Average Best Locomotion Distance ± Standard Deviation	t-statistic (against SPANN)
SPANN	4.9 ± 2.6	13.9626 ± 1.7033	-
SO-EA	0	15.7516 ± 2.9721	1.53
SO-EA	1	15.1441 ± 2.0260	1.33
SO-EA	2	16.3236 ± 2.7242	2.43
SO-EA	3	15.1532 ± 3.1696	1.05
SO-EA	4	15.2088 ± 2.2106	1.61
SO-EA	5	15.1562 ± 2.8741	1.32
SO-EA	6	16.0317 ± 2.0719	2.86
SO-EA	7	15.8033 ± 1.6159	2.07
SO-EA	8	17.4358 ± 3.2508	2.89
SO-EA	9	15.7375 ± 2.6430	1.53
SO-EA	10	16.1514 ± 2.1318	2.49
SO-EA	11	15.0614 ± 3.5612	0.86
SO-EA	12	15.4287 ± 3.0020	1.41
SO-EA	13	15.0359 ± 1.8909	1.19
SO-EA	14	16.6273 ± 2.8095	2.70
SO-EA	15	15.6150 ± 2.4605	2.58

Table 6.6: Comparison of best locomotion distance for Pareto/best solutions obtained over 10 independent runs using the SPANN algorithm and single-objective EA (SO-EA). Number of hidden units is fixed in the single-objective EA.

different at the $\alpha = 0.01$ significance level compared to SPANN while only six out of the sixteen different setups were significantly different at the $\alpha = 0.05$ significance level (number of hidden units = 2, 6, 8, 10, 14 & 15). However, it should be noted that the standard deviations in 15 out of the 16 different setups in the single-objective EA were higher than SPANN which suggests that even though the search space is much larger in the EMO case, the SPANN algorithm is still more stable in terms of its optimization results. It should also be remembered that 150 more evolutionary runs (15 setups $\times$ 10 repeats) were required in the single-objective case simply to investigate the effects of the hidden layer size on the evolution of these controllers. This would be a serious limitation for such investigations if the different number of setups required increases in magnitude (for example, consider the case where 100 or 1000 hidden units are allowed) or if the additional factors to be investigated are not discrete in nature (for example variations in the morpho-

logical parameters of the artificial creature). Moreover, in obtaining these better locomotion capabilities, there is a significant trade-off since the overall computational costs in terms of evaluating the ANN during evolution is much higher for the single-objective EA compared to SPANN (see Section 6.4.4 below).

6.4.4 Trading-Off Pareto Optimality Against Computational Cost

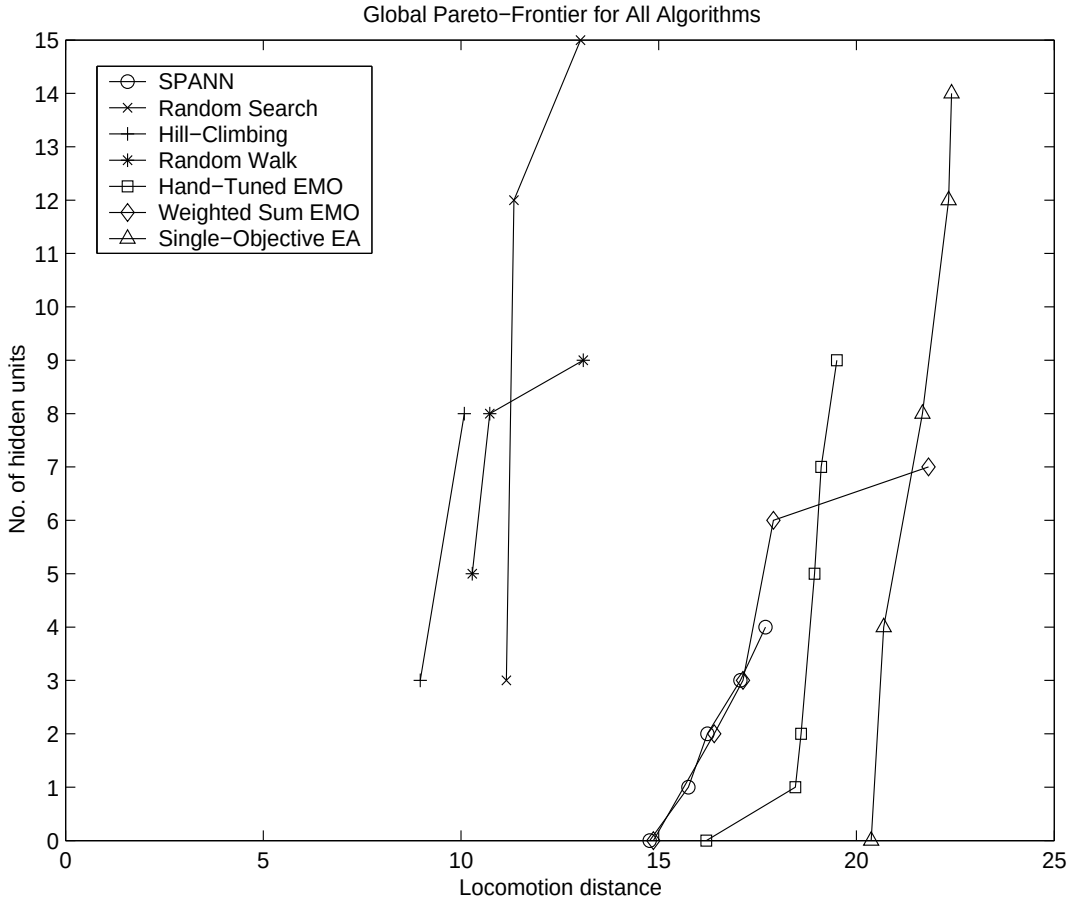


Figure 6.22: Pareto-front of solutions obtained using all algorithms over all runs conducted. X-axis: Locomotion distance, Y-axis: No. of hidden units.

Figure 6.22 plots the global Pareto-front for all the different algorithms used in evolving ANN controllers for the artificial creature. Two distinct groups of Pareto optimal solutions can be seen in this graph. One group is formed using the

random search, hill-climbing and random walk algorithms where the solutions were significantly worse than the other group of solutions formed by SPANN, the hand-tuned EMO, weighted sum EMO and single-objective EA. In the latter group, the best global Pareto-front was obtained using the single-objective EA when viewed in terms of the locomotion capability achieved, followed by the weighted sum method, the hand-tuned EMO and finally by SPANN. Although the Pareto-front of SPANN was not as optimal as the other algorithms, the controller with the highest locomotion distance discovered by each algorithm had the smallest hidden layer size for SPANN (4 nodes) compared to the hand-tuned EMO (9 nodes), weighted sum EMO (7 nodes) and the single-objective EA (14 nodes).

Algorithm	Best Locomotion Distance	$+/-\%$ of SPANN	Total Computational Cost	$+/-\%$ of SPANN
SPANN	17.6994	-	909,520,500	-
Random Search	13.0225	-26.4%	1,122,402,000	$+23.4\%$
Hill-Climbing	10.0832	-43.0%	1,047,833,000	$+15.2\%$
Random Walk	13.0900	-26.0%	1,124,060,000	$+23.5\%$
HT-EMO	19.5051	$+15.9\%$	7,529,814,400	$+727.9\%$
WS-EMO	21.8228	$+23.3\%$	3,073,867,500	$+238.0\%$
SO-EA	22.4069	$+26.5\%$	1,441,441,000,000	$+15748.4\%$

Table 6.7: Comparison of overall best locomotion controller obtained and corresponding computational cost using SPANN against all other algorithms.

Table 6.7 compares the overall best solution found by the different algorithms against the overall computation cost involved in discovering these solutions. The computational cost is estimated using the total number of hidden unit activations registered during the search process for each algorithm. This is a reasonable estimate since most of the computational time involved in conducting these experiments is spent on the evaluation of different ANN controllers by way of physically simulating the creature as guided by each newly generated controller within the Vortex physics-based world (see Section 3.1.1). Therefore, the computational cost (C) will differ between different algorithms as a function of the number of hidden unit activations required to evaluate the fitness of each newly generated genotype (A), the number of new genotypes generated per evolutionary run (G) and the number

of evolutionary runs per algorithm (R), as described by the following equation:

$$C = A \times G \times R \tag{6.4}$$

The best overall solution in terms of locomotion distance was obtained using the single-objective EA where the improvement over SPANN was 26.5%. However, the corresponding computational cost was a staggering 15,748% more than SPANN. This was mainly due to the number of repeated runs required for each different hidden layer size as well as the high usage of hidden units in the runs involving larger hidden layer sizes, which cannot be changed within each evolutionary run. Again, although better locomotion capabilities were obtained using the hand-tuned EMO and weighted sum EMO, dramatically higher computational costs were also associated with the use of the hand-tuned (727%) as well as the weighted sum methods (238%) compared to SPANN. Again these increases can be attributed to the need for repeated evolutionary runs required to test different weight assignments and crossover/mutation rates respectively in these algorithms. However, the inclusion of another objective in these two algorithms, which allowed for the minimization of the hidden layer size, did reduce the computational cost significantly compared to the single-objective EA. Although inferior results were obtained using the random search, hill-climbing and random walk algorithms, the overall computational costs were still higher than SPANN. This is due to the fact that these algorithms were only optimizing the locomotion component as in the single-objective EA and hence did not impose any pressure on minimizing the usage of hidden units during optimization.

In summary, although better controllers were evolved for locomotion distance using the single-objective EA, the corresponding trade-off in terms of overall computational cost was dramatically and unfavorably large. The trade-off between obtaining better locomotion capabilities and computational cost was again significantly and unfavorably large using the hand-tuned and weighted sum EMO algorithms. Hence, the SPANN algorithm has been shown to provide reasonably good results in terms of evolving locomotion controllers while at the same time providing the lowest overall computational cost compared to all other algorithms investigated

in this study.

6.4.5 Redundancy in Best Evolved Controllers

We now compare the amount of redundancy present in the overall best evolved controllers in terms of the hidden units as well as weight synapses obtained from SPANN against the hand-tuned, weighted sum and single-objective methodologies. For this set of experiments, we selected the overall best controller evolved for locomotion distance obtained from SPANN, the hand-tuned EMO algorithm, the weighted sum EMO algorithm and the single-objective EA as representative ANN architectures optimized using these respective approaches. These controllers, which are used in the lesioning experiments that test for redundancy in the ANN architecture, are listed in Table 6.8.

Algorithm	Locomotion Distance	No. of Hidden Units
SPANN	17.6994	4
HT-EMO	19.5051	9
WS-EMO	21.8228	7
SO-EA	22.4069	14

Table 6.8: Overall best locomotion controllers evolved using the SPANN, hand-tuned (HT-EMO), weighted sum (WS-EMO) and single-objective (SO-EA) algorithms used in the lesioning experiments.

In our analysis, redundancy is considered to be present when a controller can allow deletion of: (1) entire hidden units, or (2) individual weight synapses, without loss of fitness of more than 1 unit of locomotion distance compared to the intact controller originally evolved. The first comparison involved deletion of entire nodes in the hidden layer and can be regarded as macro-lesioning of the controller. This was done in a fashion similar to Migliono and Walker (2002) where all possible combinations of hidden units used in the ANN controller were systematically deleted. The lesioned controller is then re-evaluated and the new fitness achieved recorded. For example, if the best evolved controller had 4 hidden units, then all possible combinations of networks with 1 node lesioned are first evaluated, followed by all combinations of networks with 2 nodes lesioned and so forth terminating when

the next level of hidden unit removal causes the fitness evaluations of the lesioned controllers to fall below the redundancy threshold. The second comparison involved the deletion of individual weight synapses which can be regarded as micro-lesioning of the controller. The test for weight synapse redundancy was carried out in a greedy fashion due to the very large numbers of possible weight synapse combinations: first find the least loss of locomotion capability with 1 weight synapse lesioned, then proceed to find the next least loss of locomotion capability with another weight synapse lesioned by keeping the weight synapse found in the preceding step lesioned, and so forth terminating when the next level of weight synapse removal causes the fitness evaluations of the lesioned controllers to fall below the redundancy threshold. This second redundancy test is less drastic compared to the first test since the deletion of a single hidden node would cause entire sets of weight synapses to be also deleted in a single step. As such, the second redundancy test of lesioning only at the weight synapse level allows for a finer investigation into the controller's evolved architecture.

6.4.5.1 Hidden Unit Redundancy

Algorithm	Redundancy Fitness Threshold	Best Lesioned Fitness	Worst Lesioned Fitness	Average Lesioned Fitness ± Standard Deviation
SPANN	16.6994	12.8242	8.6991	10.9156 ± 1.9204
HT-EMO	18.5051	*18.6472*	1.6899	10.8691 ± 7.0303
WS-EMO	20.8228	18.6352	8.5114	14.4979 ± 3.8776
SO-EA	21.4069	17.5729	2.1421	12.9751 ± 4.8180

Table 6.9: Comparison of locomotion distance of overall best controller evolved using the SPANN, hand-tuned (HT-EMO), weighted sum (WS-EMO) and single-objective (SO-EA) algorithms with 1 hidden node lesioned.

None of the overall best controllers evolved for locomotion distance using SPANN, weighted sum and single-objective methodologies showed any redundancy in terms of hidden units present in the ANN controller. All possible combinations of controllers with a single hidden node removed from the optimized architecture using

these algorithms resulted in locomotion fitness below the redundancy threshold as shown in Table 6.9. However, the overall best controller evolved using the hand-tuned EMO algorithm did have one redundant hidden unit (the eighth node) which could be lesioned without causing the controller's capabilities to fall below the fitness threshold. This phenomenon together with the results from further levels of hidden unit lesioning of the overall best controller evolved using the hand-tuned EMO algorithm are discussed in the next paragraph. Surprisingly, the lesioning of a single hidden unit appeared to also have the most detrimental effect on the best controller evolved using the hand-tuned EMO algorithm in terms of the average loss of locomotion fitness compared to all other algorithms. Furthermore, the lesioning of a particular hidden node in the best controller evolved using the hand-tuned EMO algorithm also produced the worst 1-node lesioned controller, which only achieved a minuscule locomotion distance of just over 1.6 units. The removal of entire hidden nodes from the optimized ANN controllers seemed to result in large scale loss of locomotion capability suggesting that macro-lesioning of these evolved architectures is too drastic due to removal of not only a single hidden node but an entire set of weight synapses connecting to and originating from the lesioned hidden node. If the redundancy test were to be concluded at this coarse level, then the results would have indicated that no redundancy was present at all in the evolved controllers obtained using SPANN, the weighted sum EMO algorithm and the single-objective EA. However, a redundancy test at the finer weight synapse level, which is presented in the next section, showed otherwise.

Algorithm	Redundancy Fitness Threshold	Best Lesioned Fitness	Worst Lesioned Fitness	Average Lesioned Fitness ± Standard Deviation
HT-EMO	18.5051	18.1569	1.3326	9.4544 ± 5.6139

Table 6.10: Locomotion distance of overall best controller evolved using the hand-tuned EMO (HT-EMO) algorithm with 2 hidden nodes lesioned.

Before proceeding with the analysis at the weight synapse level, we first discuss the results obtained from the lesioning of different combinations of 2 hidden nodes from the overall best controller evolved using the hand-tuned EMO algorithm

since lesioning of 1 hidden node did show redundancy in this particular controller. None of the controllers which had 2 hidden nodes removed could produce locomotion fitness above the redundancy threshold as shown in Table 6.10. Compared to the overall best controller evolved using the self-adaptive SPANN algorithm, the hand-tuned EMO algorithm evolved a controller with more redundancy at the hidden unit level. Hence, the self-adaptive crossover and mutation rates appeared to have benefited the Pareto evolutionary optimization process in that SPANN was able to find a more compact network with less redundancy compared to the hand-tuned algorithm, which had pre-determined and fixed crossover and mutation rates during the optimization process.

6.4.5.2 Weight Synapse Redundancy

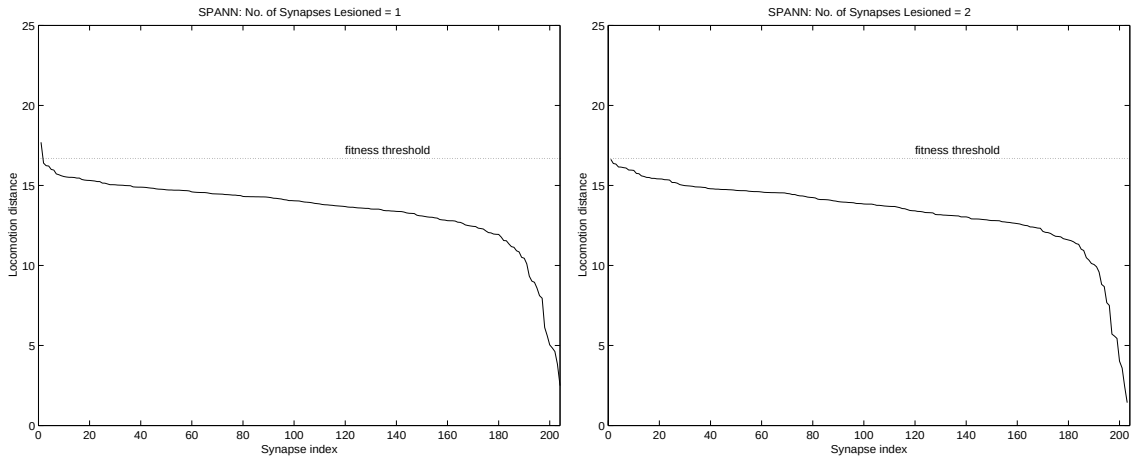


Figure 6.23: Locomotion distance for overall best controller evolved using SPANN with lesioning of 1. 1 weight synapse (left), 2. 2 weight synapses (right). The index assigned to the lesioned weight synapse is labelled according to the order of lesioning. X-axis: Synapse index, Y-axis: Locomotion distance.

Figure 6.23 illustrates the loss of locomotion fitness as weight synapses were lesioned in the overall best controller evolved using SPANN. There was only one particular weight synapse that could be lesioned without causing the controller's performance to fall below the fitness threshold (Figure 6.23.1). No other lesioning of

a single synapse could produce a fitness above the redundancy threshold. Additionally, no controller with 2 synapses lesioned could produce controllers that maintained their performance above the fitness threshold (Figure 6.23.2). This meant that the best evolved controller from SPANN only had a redundancy of one weight synapse and furthermore this occurred only with a specific weight synapse, which was the connection between the input from the joint sensor that measures the angle between the torso and the upper back left limb (Sensor x_1) and the first node in the hidden layer of the ANN.

Figure 6.24 illustrates the loss of locomotion fitness as weight synapses were lesioned in the overall best controller evolved using the hand-tuned EMO algorithm. This controller exhibited a high level of weight synapse redundancy where up to 281 synapses could be removed without causing the controller to fall below the fitness threshold. No controller with 282 synapses lesioned could produce controllers that maintained their performance above the fitness threshold (Figure 6.24.4). In line with results obtained from macro-lesioning at the hidden unit level, a much higher level of weight synapse redundancy should be expected in this controller compared to the overall best controllers evolved using the other algorithms. This is the case since the analysis from the previous section showed that an entire hidden node could be removed without causing the controller to fall below the fitness threshold, which correspondingly means that the entire set of synapses connected to and originating from this particular hidden unit were redundant. Thus, at least 30 weight synapses that are connected to this hidden unit can be removed without causing the lesioned controller to fall below the fitness threshold. The number of different weight synapses that could be removed varied considerably at different levels of lesioning. For example, Figure 6.24.2 showed that only approximately 25 different synapses could be removed at the 100-synapse level without causing the controller's locomotion capabilities to fall below the redundancy threshold. On the other hand, Figure 6.24.3 showed that approximately 75 different synapses could be removed at the 200-synapse level without causing the lesioned controller to fall below the fitness threshold. The fluctuations observed with regards to the number of different

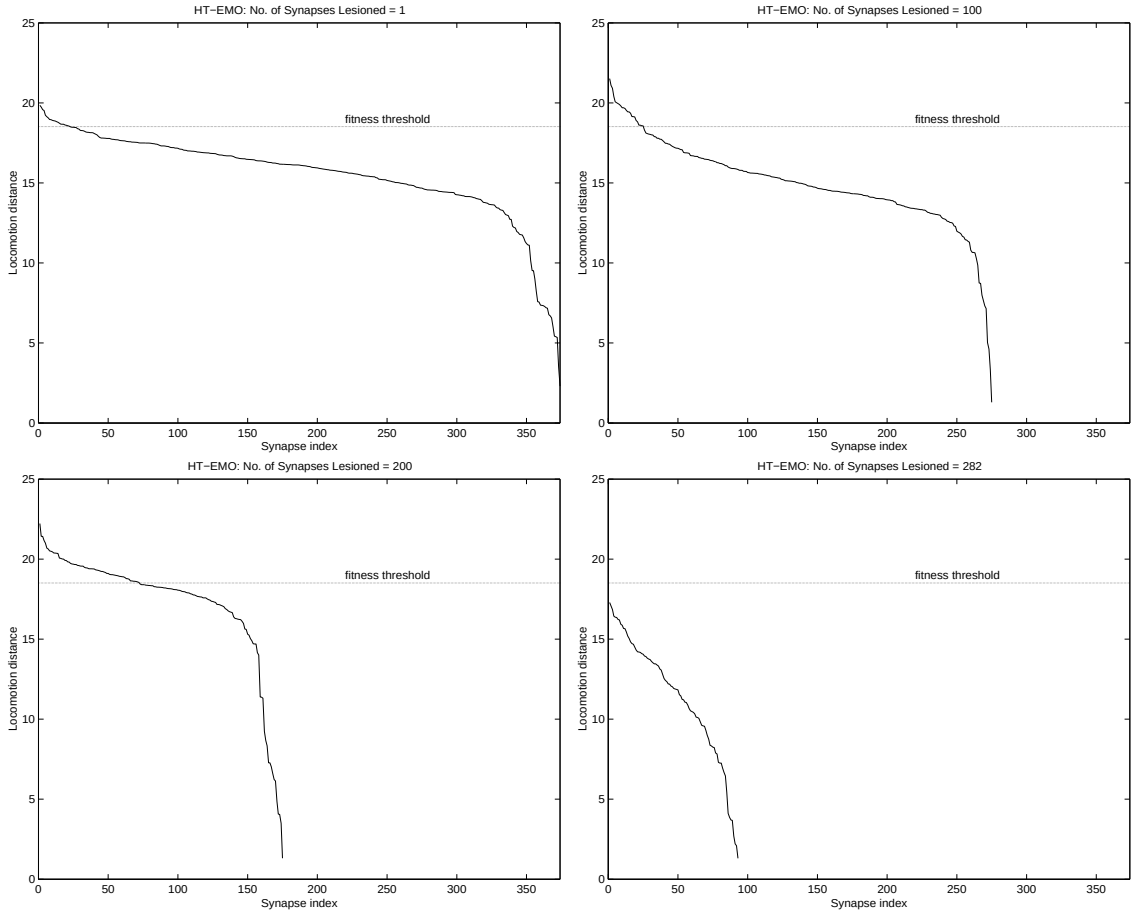


Figure 6.24: Locomotion distance for overall best controller evolved using the hand-tuned EMO algorithm (HT-EMO) with lesioning of 1. 1 weight synapse (top left), 2. 100 weight synapses (top right), 3. 200 synapses (bottom left), 4. 282 synapses (bottom right). The index assigned to the lesioned weight synapse is labelled according to the order of lesioning. X-axis: Synapse index, Y-axis: Locomotion distance.

synapses that could be removed at various levels of weight synapse lesioning are most probably due to the greedy nature in which the synapse lesioning takes place combined with the complex dynamics that takes place within the evolved ANN. A weight synapse lesioned presently will provide the best performance at the current level but the effects of this lesioning may at certain stages become highly sensitive to further lesioning and vice versa due to the combinatorial effects that occur between different weight synapses and hidden nodes in the network, which cannot be

ascertained by this one-step lookahead algorithm.

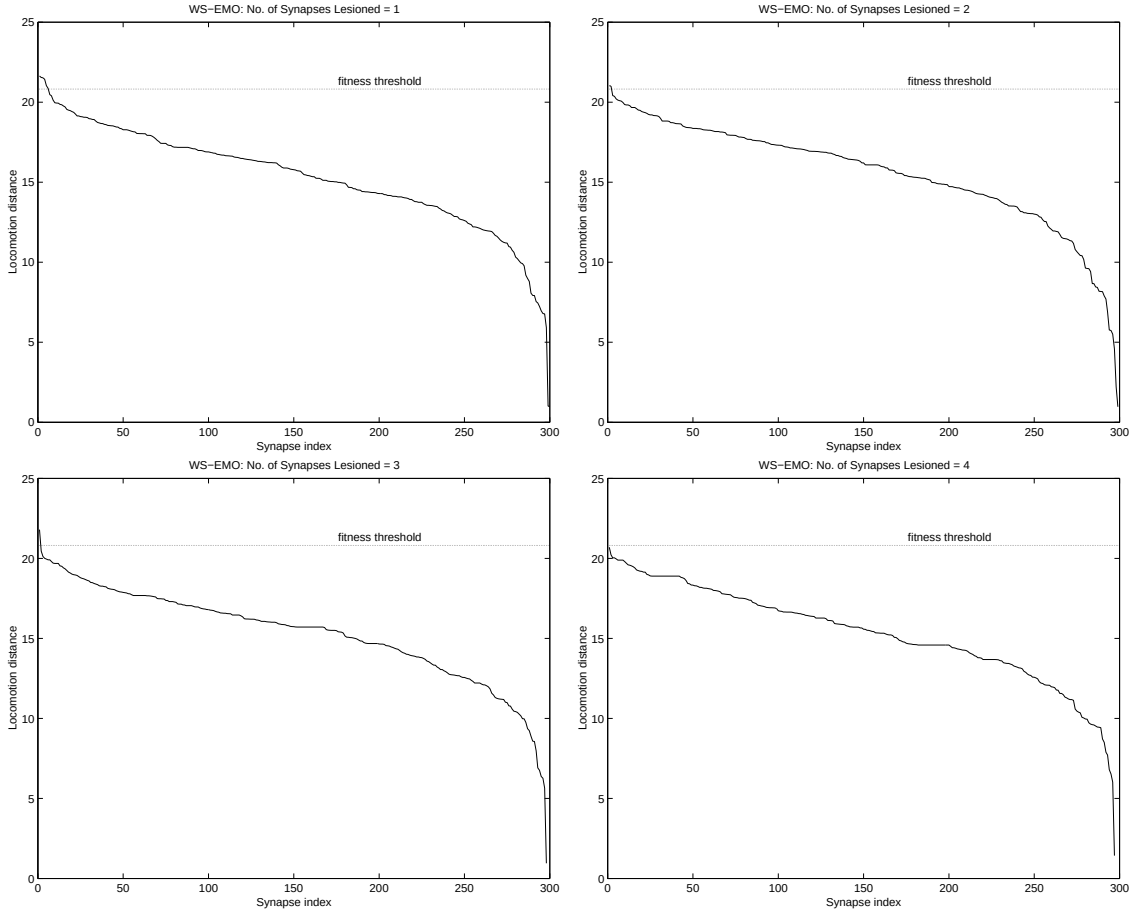


Figure 6.25: Locomotion distance for overall best controller evolved using the weighted sum EMO algorithm (WS-EMO) with lesioning of 1. 1 weight synapse (top left), 2. 2 weight synapses (top right), 3. 3 synapses (bottom left), 4. 4 synapses (bottom right). The index assigned to the lesioned weight synapse is labelled according to the order of lesioning. X-axis: Synapse index, Y-axis: Locomotion distance.

Figure 6.25 illustrates the loss of locomotion fitness as weight synapses were lesioned in the overall best controller evolved using the weighted sum EMO algorithm. Up to three synapses could be lesioned without causing the controller to fall below the fitness threshold. No controller with 4 synapses lesioned could produce controllers that maintained their performance above the fitness threshold (Figure 6.25.4). In terms of the number of different weight synapses that could

be removed and still produced controllers that performed above the threshold, 6 controllers were found when 1 synapse was lesioned (Figure 6.25.1), followed by 2 controllers when 2 synapses were lesioned (Figure 6.25.2) and finally by only 1 controller when 3 synapses were lesioned (Figure 6.25.3). Hence there was more synaptic redundancy present in the overall best controller evolved using the weighted sum method compared to SPANN.

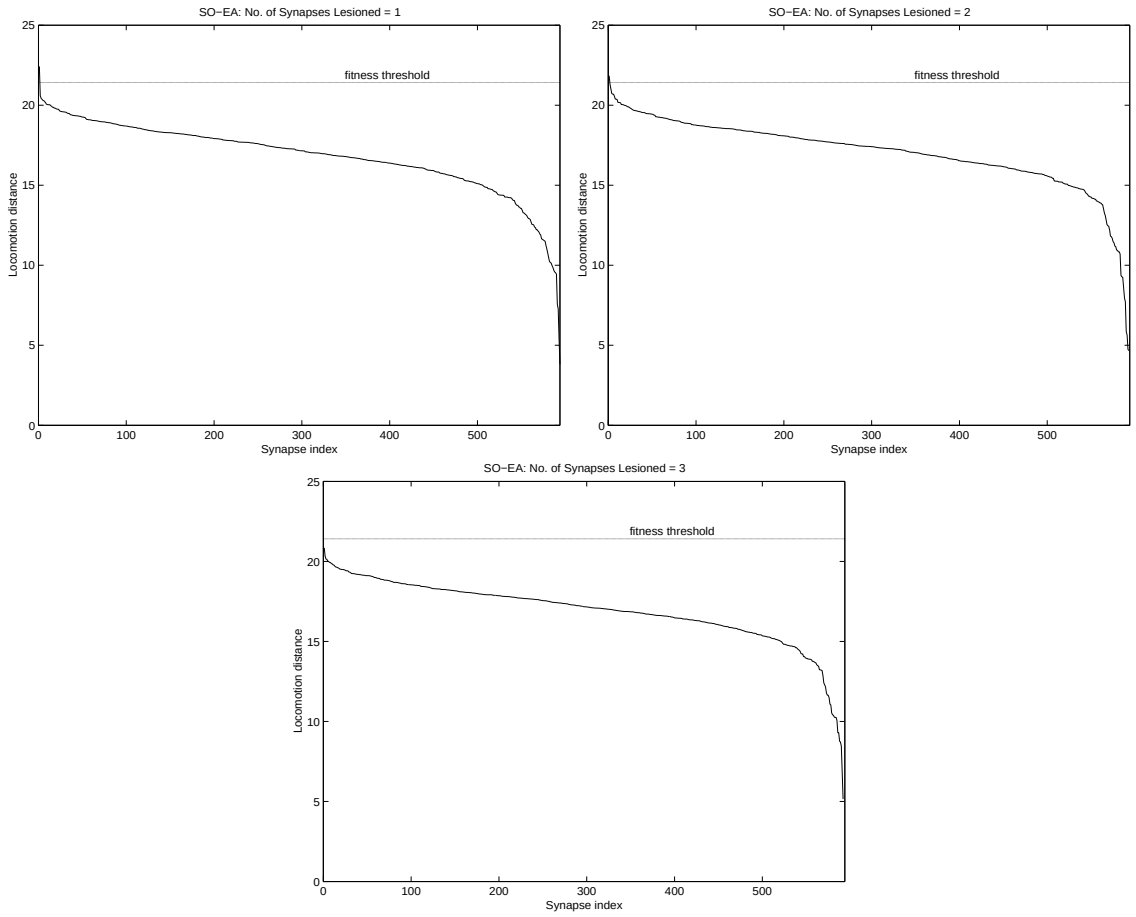


Figure 6.26: Locomotion distance for overall best controller evolved using the single-objective EA (SO-EA) with lesioning of 1. 1 weight synapse (top left), 2. 2 weight synapses (top right), 3. 3 synapses (bottom). The index assigned to the lesioned weight synapse is labelled according to the order of lesioning. X-axis: Synapse index, Y-axis: Locomotion distance.

Figure 6.26 illustrates the loss of locomotion fitness as weight synapses

were lesioned in the overall best controller evolved using the single-objective EA. Up to 2 synapses could be lesioned without causing the evolved controller to fall below the fitness threshold. No controller with 3 synapses lesioned could produce controllers that maintained their performance above the fitness threshold (Figure 6.26.3). Only a particular weight synapse could be removed when 1 synapse was lesioned which produced a controller that performed above the threshold (Figure 6.26.1). Following this lesioning, only another specific weight synapse could be removed at the 2-synapse lesioning level that resulted in a controller that performed above the threshold (Figure 6.26.2). Thus, there was also more synaptic redundancy present in the overall best controller evolved using the single-objective EA compared to SPANN but less synaptic redundancy compared to the hand-tuned and weighted sum method. A summary comparing the number of redundant weight synapses present in the best evolved controller obtained from SPANN against those obtained using the hand-tuned, weighted sum and single-objective algorithms is given in Table 6.11.

Algorithm	No. of Redundant Synapses
SPANN	1
HT-EMO	281
WS-EMO	3
SO-EA	2

Table 6.11: Number of redundant synapses in the best evolved controllers from the SPANN, hand-tuned (HT-EMO), weighted sum (WS-EMO) and single-objective (SO-EA) algorithms.

6.4.6 SPANN Against NSGA-II

To conclude our verification of SPANN as a beneficial Pareto EMO algorithm, we compare its results against one of the current state-of-the-art Pareto EMO algorithms called NSGA-II (Deb, Agrawal, Pratab, and Meyarivan 2000). Our objective here is simply to verify that the solutions obtained using SPANN are comparable to those obtained using a well-known and well-tested Pareto EMO algorithm

and not to determine which Pareto EMO algorithm is better for evolving locomotion controllers as this is beyond the scope of this thesis. The NSGA-II algorithm was obtained from the authors' web site (KanGAL 2003) and used as a benchmark algorithm without any modification to the NSGA-II algorithm. The NNType3 architecture was again used in this set of experiments and all other parameters remained the same: 1000 generations, 30 individuals, 500 timesteps and 10 repeated runs. NSGA-II requires a number of other parameters to be set by the user including the crossover and mutation rates which are non-self-adaptive. Recently, the authors of NSGA-II conducted a comprehensive comparative study of NSGA-II against other EMO algorithms, which were reported in Deb, Pratab, Agrawal, and Meyarivan (2002). Hence, in the first setup, these user-defined parameters were set according to those used in the above-mentioned comparative study as follows: crossover rate 90%, mutation rate for real-coded variables 0.1553% (representing the reciprocal of the number of real-coded variables), and mutation rate for binary-coded variables 6.6667% (representing the reciprocal of the number of binary-coded variables), distribution index for crossover operator 20, distribution index for mutation operator 20, and single-point crossover.

Algorithm	Overall Best Locomotion Distance	Average Best Locomotion Distance ± Standard Deviation	t-statistic (against SPANN)	No. of Hidden Units
SPANN	17.6994	13.9626 ± 1.7033	-	4.9 ± 2.6
NSGA-II Setup 1	15.5452	11.7421 ± 2.0497	(3.78)	0 ± 0
NSGA-II Setup 2	18.3941	16.2022 ± 1.5860	2.85	6.8 ± 2.3
NSGA-II Setup 3	20.4144	17.8635 ± 1.9744	4.54	8.4 ± 2.1
NSGA-II Setup 4	20.9806	16.2667 ± 2.1868	2.54	7.7 ± 1.7

Table 6.12: Comparison of best locomotion distance for Pareto solutions obtained over 10 independent runs using the SPANN and NSGA-II algorithms. Setup 1: c=90%, m(r)=0.1553%, m(b)=6.6667%. Setup 2: c=50%, m(r,b)=50%. Setup 3: c=50%, m(r,b)=90%. Setup 4: c=90%, m(r,b)=50%. **c** = crossover rate, **m(r)** = mutation rate for real-coded variables, **m(b)** = mutation rate for binary-coded variables.

Table 6.12 lists the best Pareto solutions for locomotion distance obtained using the NSGA-II algorithm and compares them against those obtained using the SPANN algorithm. The best solutions obtained using the first setup for NSGA-II produced controllers that used no hidden units in all 10 runs. The overall best locomotion distance achieved was lower than that obtained using SPANN. The very small mutation rate used in this setup most probably caused the evolutionary search to prematurely converge to local optima centered around controllers which did not use any hidden units. A t-test showed that the results obtained using NSGA-II were significantly worse than SPANN at the $\alpha = 0.01$ significance level for this setup. Also, the overall best controller from SPANN achieved over 2 units of distance more than the overall best controller obtained from this setup of NSGA-II (representing a decrease of 12.2% compared to the overall best locomotion distance achieved by SPANN).

To overcome the inferior results obtained using the setup reported in (Deb, Pratab, Agrawal, and Meyarivan 2002), a second experiment utilizing the best combination of crossover and mutation rates obtained from the hand-tuned EMO was conducted. This second setup used crossover and mutation rates of 50% with all other parameters unchanged. Much better results were obtained in this second setup, where the overall best controller in terms of locomotion achieved a higher distance than that obtained using SPANN by just under 0.7 units (representing a 3.9% improvement over the best locomotion distance achieved by SPANN). A t-test showed that the solutions obtained using NSGA-II with the second setup were significantly better than those obtained using SPANN at the $\alpha = 0.05$ significance level.

Since these results suggest that a high mutation rate may improve the performance of NSGA-II, we carried out a third experiment using a setup with an even higher mutation rate of 90% while maintaining the crossover rate at 50%. However, a t-test comparing the results from this third setup against the second setup for NSGA-II showed no significant improvements. To test whether a higher crossover rate would yield better results, a fourth experiment was conducted using

a setup with crossover rate of 90% and maintaining the mutation rate at 50%. Again, a t-test showed no significant improvements in the results obtained with the fourth setup compared to the second setup of NSGA-II. The solutions obtained with third and fourth setup for NSGA-II were significantly better than those obtained with SPANN at the $\alpha = 0.01$ and $\alpha = 0.05$ levels respectively. The best solutions obtained from the third setup of NSGA-II used an average of 8.4 hidden units, which is almost double the number used by the best solutions obtained using SPANN, while the fourth setup used an average of 7.7 hidden units.

The Pareto-frontiers obtained over the 10 runs of NSGA-II for the four setups are depicted in Figure 6.27. The first setup only produced controllers that did not make use of any hidden units in the ANN controller as can be seen in Figure 6.27.1. All runs converged to solutions that did not require any hidden layer transformation resulting in purely reactive controllers being generated. In comparison, the second, third and fourth setups which used much higher mutation rates all produced a much greater variety of controllers as shown by the Pareto-fronts plotted in Figures 6.27.2, 6.27.3 and 6.27.4 respectively.

Figure 6.28 plots the global Pareto-front of SPANN and NSGA-II. It can be seen that the Pareto-front generated through 10 runs of SPANN is comparable though dominated by the Pareto-front generated through 40 runs of NSGA-II (10 runs each in Setup 1–4). The solution with 0 hidden units of the NSGA-II global Pareto-front was contributed from the first setup of NSGA-II while the remaining 8 other solutions on the global Pareto-front were contributed from the other three setups.

In summary, as with the hand-tuned EMO algorithm, there is a trade-off between obtaining better locomotion controllers using NSGA-II at the cost of incurring greater computational expense to find the optimal parameter settings. It is clear that the performance of NSGA-II is also sensitive to the parameters used. As future work, it would be interesting to implement a self-adaptive version of NSGA-II for a direct comparison against the self-adaptive SPANN.

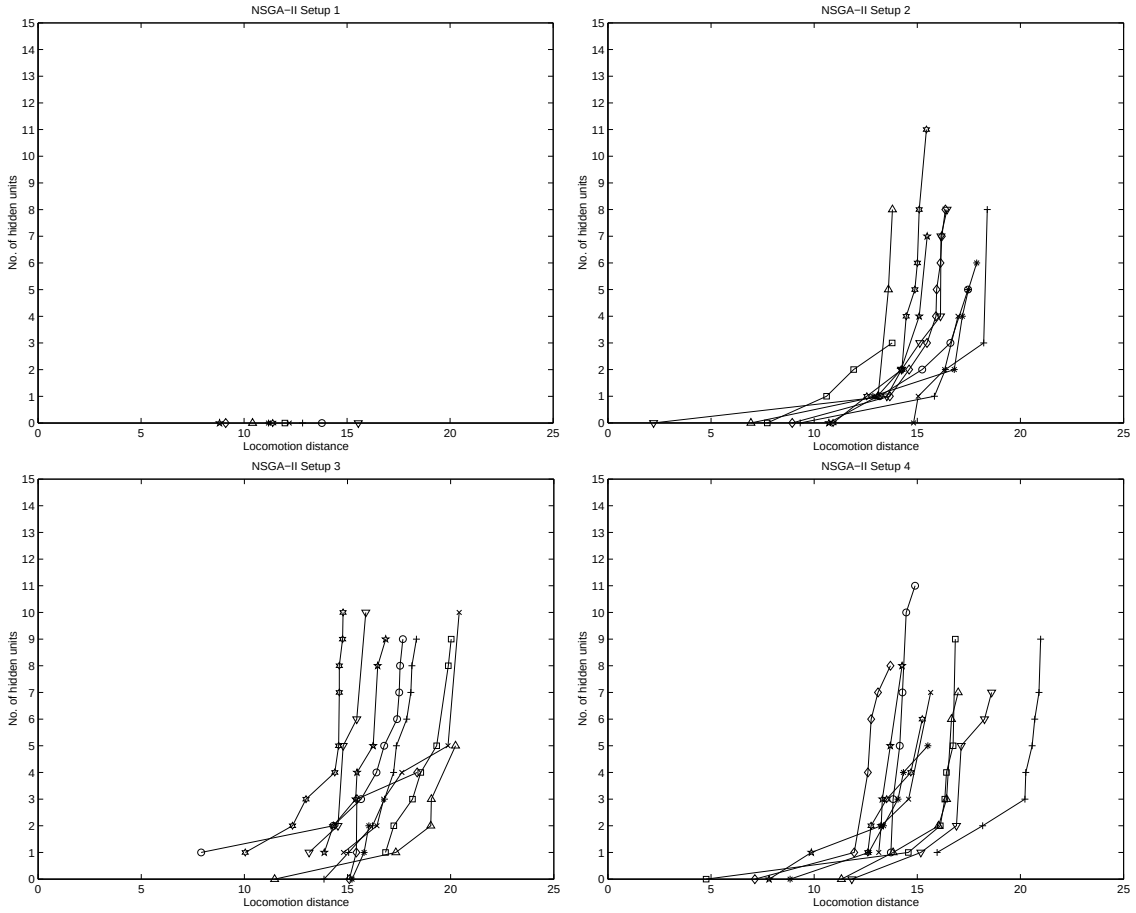


Figure 6.27: Pareto-front of solutions obtained for 10 runs using the NSGA-II algorithm for Setup 1 (top left), Setup 2 (top right), Setup 3 (bottom left), Setup 4 (bottom right). Setup 1: c=90%, m(r)=0.1553%, m(b)=6.6667%. Setup 2: c=50%, m(r,b)=50%. Setup 3: c=50%, m(r,b)=90%. Setup 4: c=90%, m(r,b)=50%. **c** = crossover rate, **m(r)** = mutation rate for real-coded variables, **m(b)** = mutation rate for binary-coded variables. X-axis: Locomotion distance, Y-axis: No. of hidden units.

6.5 Chapter Summary

The self-adaptive Pareto SPANN algorithm was compared against EMO algorithms that utilized hand-tuning of crossover/mutation rates and weighted sum approaches as well as a single-objective EA. It was found that SPANN discovered reasonably good quality controllers but most importantly required significantly less

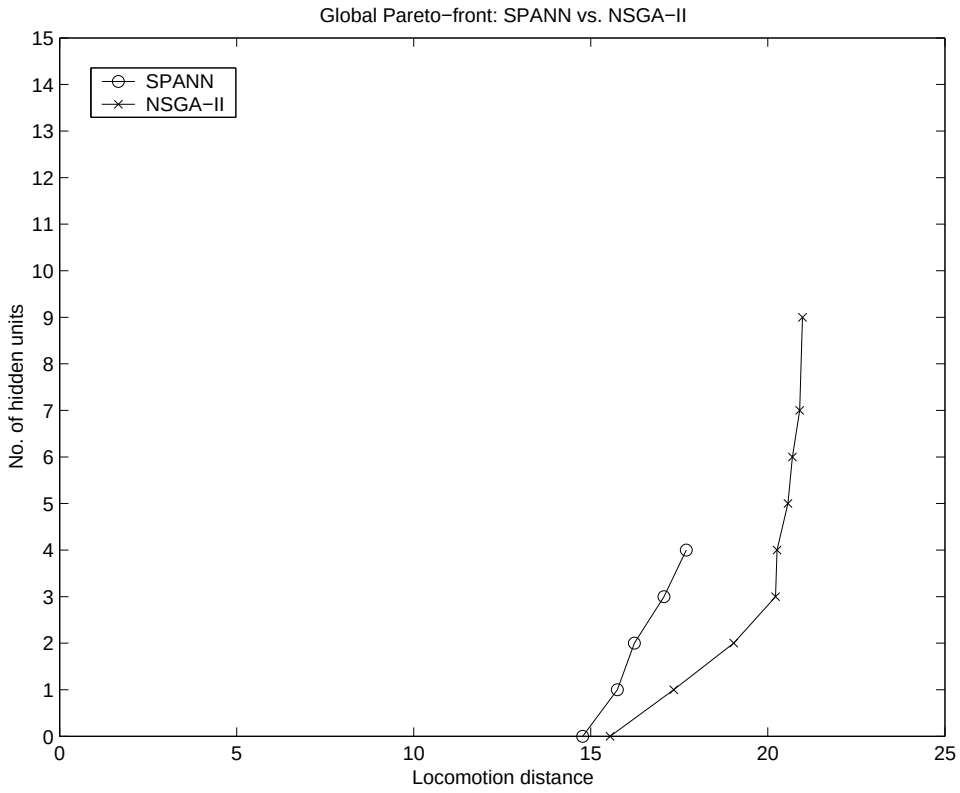


Figure 6.28: Global Pareto-front of controllers obtained using the SPANN and NSGA-II algorithms. X-axis: Locomotion distance, Y-axis: No. of hidden units.

overall computational costs. Although better solutions were found using the single-objective EA, the weighted sum and hand-tuned EMO algorithms, the trade-off in terms of computational costs was extremely high in comparison to SPANN. Furthermore, there was more redundancy present in the best controllers evolved using the hand-tuned, weighted sum and single-objective methodologies compared to the self-adaptive Pareto approach. The performance of SPANN was also found to be comparable to that of a current state-of-the-art benchmark Pareto EMO algorithm, NSGA-II. Therefore, the self-adaptive Pareto SPANN algorithm has been shown to be a highly beneficial EMO algorithm to use for evolving artificial creature controllers compared to other more conventional evolutionary optimization algorithms. In the next chapter, we will present a multi-objective view towards capturing and characterizing the complexities of evolved controllers using the SPANN algorithm.

Chapter 7

Creature Complexity

[1] Complexity has been and will remain a debatable concept. We all know it when we see it, yet when we need to provide a functional definition of complexity, it becomes a mythical entity. Although the study of complex systems has attracted much interest over the last decade and a half, the definition of what makes a system complex is still the subject of much debate among researchers (Adami 1998; Feldman and Crutchfield 1998b; Standish 2001). What is complexity? Is there a universal measure of complexity? Does complexity arise when a system reaches a critical point or is there a phase transition between simple and complex systems? Are adaptability, adaptation, emergence, hierarchy, bifurcation, and self-organization evidence for complexity? These are all common concerns raised by researchers when speaking of complex systems and complexity itself.

Although there are numerous methods available in the literature for measuring complexity (Badii and Politi 1997; Edmonds 1999), it has been argued however that such complexity measures are typically too difficult to compute to be of use for any practical purpose or intent (Shalizi 2001). This chapter attempts to unfold some mysteries about complexity and to pose EMO as a simple and highly accessible methodology for characterizing the complexity of artificially evolved creatures using a multi-objective methodology. One of the main objectives of evolving

[1] Some of the material presented in this chapter have been previously published in Teo, Nguyen, and Abbass (2003).

artificial creatures is to synthesize increasingly complex behaviors and/or morphologies either through evolutionary or lifetime learning (Pfeifer and Scheier 1999; Nolfi and Floreano 2000; Hornby and Pollack 2001a; Komosinski and Rotaru-Varga 2001; Bongard 2002b). Needless to say, the term "*complex*" is generally used very loosely since there is currently no general method for comparing between the complexities of these evolved artificial creatures' behaviors and morphologies. As such, without a quantitative measure for behavioral or morphological complexity, an objective evaluation between these artificial evolutionary systems becomes very hard and typically ends up being some sort of a subjective argument.

We first present attempts at defining complexity followed by a comprehensive review of the different views of complexity from the social sciences to concrete measures in information systems and the physical sciences. Then, we propose a characterization of the notion of complexity in embodied cognition using multi-objectivity as a natural and theoretically-founded paradigm in mathematics. Specifically, we will attempt to characterize the behavioral and morphological complexities of different artificial creatures using the multi-objective controller evolution approach introduced in Chapter 5.

7.1 Complexity Defined?

The following list provides some suggested definitions of complexity and complex systems:

- "The complexity of a system S is a contingent property, depending upon the nature of the observables describing S, and their mutual interactions." — Casti (1986, p.155)
- "Complexity is the study of the behavior of macroscopic collections of such units that are endowed with the potential to evolve over time." — Coveney and Highfield (1995, p.7)
- "...a "theory of complexity" could be viewed as a theory of modelling, en-

compassing various reduction schemes (elimination or aggregation of variables, separation of weak from strong couplings, averaging over subsystems, evaluating their efficiency and, possibly, suggesting novel representations of natural phenomena." — Badii and Politi (1997, p.6)

- "I use *complex* and *complexity* intuitively to describe self-organized systems that have many components and many characteristic aspects, exhibit many structures in various scales, undergo many processes in various rates, and have the capabilities to change abruptly and adapt to external environments." — Auyang (1998, p.13)

- "Complexity is that property of a model which makes it difficult to formulate its overall behavior in a given language, even when given reasonably complete information about its atomic components and their inter-relations." — Edmonds (1999, p.72)

- "...in defining complexity we need to consider both functions of perception and analysis. For what we want to know is not whether a simple or short description can be found of every detail of something, but merely whether such a description can be found of those features in which we happen to be interested." — Wolfram (2002, p.557).

It is surprising to note that although there is a large body of literature that discusses issues relating to complexity, few actually provide a definition to complexity as used in their respective contexts (Feldman and Crutchfield 1998b). As pointed out in the introduction to this chapter, the task of defining complexity is difficult in itself, which may explain why the term "complexity" is so commonly used without qualification. A number of books authored about complexity theory confirms this observation, where an enormous range of views were drawn about what complexity means to different researchers and to different disciplines (Lewin 1993; Waldrop 1994; Mainzer 1997).

In the social sciences, complex systems typically refer to social phenomena that exhibit some form of dynamic nonlinear behavior that are difficult to explain

using basic linear models. Examples of complex systems described in the social sciences include fluctuations of stock markets and exchange rates, impacts of economic policies, human population growth and migration, societal organization, political revolutions, organizational cooperation and conflict, human interactions and their communication structures (Coveney and Highfield 1995). With reference to complex systems and the evolution of human society, Mainzer (1997) states that

> "The crucial point of the complex system approach is that from a macroscopic point of view the development of political, social, or cultural order is not only the sum of single intentions, but the collective result of nonlinear interactions." (p.253)

Biological organisms and processes associated with living things and life in general typically correspond to what we intuitively know as objects and systems that exhibit the highest degree of complexity (Badii and Politi 1997). Examples of complex systems in the biological sciences include genomic evolution, genetic regulatory networks, population ecology, morphogenesis and biological neural networks to name but a few (Mainzer 1997). More specific studies of biological complex systems have been conducted on the self-replication of bio-molecules, in-vitro RNA evolution, self-regulation of the glycolytic process, self-organization of slime moulds, pattern formation of the hydra, oscillations of the heart, spatio-temporal organization of *hymenoptera* colonies, punctuated equilibrium as catastrophe theory, planetary self-regulation and *Gaia* theory (Coveney and Highfield 1995).

In chemical and physical systems, complexity is often attributed to processes that exhibit some form of aperiodicity or possess high degrees of freedom (Badii and Politi 1997). Terms commonly associated with complex physical and chemical systems include chaos, phase transitions, bifurcation, self-organized criticality and percolation. Examples of such complex phenomena include cement formation caused by the percolation of water through irregularly shaped particles in cement powder, criticality of sand-piles that produce avalanches beyond a certain geometric configuration, fluid instabilities that produce turbulence, optical instabilities in lasers that produce quasi-periodic light patterns and fluctuations of spin-glasses

caused by the disorderly arrangements of spinning electrons (Coveney and Highfield 1995; Badii and Politi 1997; Adami 1998).

As can be seen from the diverse interpretations and applications to social, biological and physical systems, complexity provides very different meanings depending on the context it is being referred to. Perhaps Mainzer (1997) provides the best summary of what complex systems encompasses:

> "...the theory of nonlinear complex systems cannot be reduced to special natural laws of physics ...it is an interdisciplinary methodology to explain the emergence of certain macroscopic phenomena via the nonlinear interactions of microscopic elements ..." (p.1)

In the next section, we will provide more concrete examples of complex systems and specifically what types of measures have been formulated to capture complexity across the different disciplines.

7.2 Measures of Complexity

In this section, we present a review of existing measures for defining or simply characterizing complexity as viewed from the social, physical and biological sciences' perspectives. A summary of the literature surveyed on general reviews and specific applications of complexity measures is given in Table 7.1. Here we give a high-level survey of the more significant measures from these diverse fields. Our intention is simply to provide an indication of the wide spectrum of efforts in trying to capture complexity into something mathematical or formal so as to assist with the characterization or comparison of different systems. A more comprehensive and detailed survey of existing complexity measures is available elsewhere (Edmonds 1999). Nonetheless, our shorter survey will show that such measures are often highly specific, being specially designed or formulated for application in a particular domain or area of research not readily transferable to another application domain. A discussion of the advantages and disadvantages associated with these different methodologies for measuring complexity will be also highlighted. We will also

Category	Reference
General Reviews	Casti (1986)
	Lewin (1993)
	Waldrop (1994)
	Coveney and Highfield (1995)
	Mainzer (1997)
	Auyang (1998)
Biological Sciences	Cavalier-Smith (1985)
	Bonner (1988)
	Atlan and Koppel (1990)
	Smith (1994)
	Maynard Smith and Szathmary (1995)
	Nehaniv (2000a), Nehaniv (2000b)
	Szathmary, Jordan, and Pal (2001)
Social Sciences	Kelly (1955)
	Albin (1980)
	Cooper (1993)
	Holm (1993)
	Lyon (1993)
	Gibson (1998)
	Halford, Wilson, and Phillips (1998)
	Clement (1999)
	Neyman and Okada (1999)
	Butts (2001)
	Andrews and Halford (2002)
	DeShazo and Fermo (2002)
	Warren and Gibson (2002)
Physical Sciences	Shannon (1948)
	Kolmogorov (1965)
	Bennett (1988)
	Badii and Politi (1997)
	Wolpert and MacReady (1997)
	Adami (1998)
	Feldman and Crutchfield (1998a), Feldman and Crutchfield (1998b)
	Edmonds (1999)
	Shalizi (2001)
	Standish (2001)
	Wolfram (2002)

Table 7.1: Summary of literature survey on reviews of complexity measures and their applications in the biological, social and physical sciences.

cover the more recent complexity measures suggested since the review conducted by Edmonds (1999) especially for biological organisms as they may provide critical insights to the measurement of complexity in their artificial counterparts.

7.2.1 Social Sciences

An early attempt to capture complexity in a numerical form exists in the psychology literature. It is called cognitive complexity and is used to describe the complexity of mental constructions of the world possessed by an individual (Kelly 1955). Cognitive complexity here is estimated numerically by counting the number of different relationships constructed by the subject from given object attributes. In this sense, a person who sees the world in more dimensions would be considered to have a higher cognitive complexity. A related and more recent technique for measuring cognitive complexity called Relational Complexity (RC) theory was proposed by Halford, Wilson, and Phillips (1998). It is defined as the number *(arity)* of relations between entities or arguments in a given decision task. For example, an unary relation would have one entity, such as *woman(Jane)* and a binary relation would have two entities such as *married(Jane,Dean)*. Hence, each entity corresponds to a variable or attribute and an n-ary relation maps to a set of points in an n-dimensional cognitive space. A recent study has tested the validity of this metric and was found to be effective in measuring the cognitive development of young children (3–8 years) (Andrews and Halford 2002).

In studies of group and organizational behavior, an early conceptualization of measuring the complexity of social interactions was proposed by Albin (1980). It applies graph theory to participating individuals within an interacting group and measures the level of complexity of social actions based on the connectivity of actions between individuals. In a more recent application of complexity to the social sciences, Butts (2001) proposed the use of algorithmic complexity to measure the complexity of social networks. Again based on graph theory, the interaction of roles representative of human social structures are first represented as directed graphs, the complexity of which is then measured according to the amount of "reducibility" or "compressibility" that can be achieved on the network. In this case, a social network with higher compressibility would be considered to have higher reducibility and the converse in a social network with low reducibility. Interestingly, complexity has also been applied to command and control theory associated with military operations

in an attempt to unify difficulties encountered with such systems (Cooper 1993). Specifically, three types of complexity were identified: dimensional, uncertainty and computational, although no actual method of measuring these complexities within the military setting was given.

Another area where complexity has been applied numerically is the study of human language processing. In a theory called Syntactic Prediction Locality Theory (SPLT) proposed by Gibson (1998), the complexity of a sentence can be predicted according to the memory cost associated with keeping a partial sentence in memory and integration cost associated with integrating new words into existing syntactical structures built thus far. Memory cost is quantified according to the number of syntactic categories necessary to complete the current input string as a grammatical sentence. Integration cost is quantified according to the distance between an incoming input and the nearest syntactic component it attaches to. This technique has been used to empirically measure how different localizations of noun phrases affected sentence complexity (Warren and Gibson 2002).

Complexity in economics typically refers to simply the relaxation of assumptions made on the behavior of market agents (Edmonds 1997). More specific applications of complexity can be found in game theory where the number of agent states is used as a measure of economic complexity (Holm 1993). An entropy-based measure has been formulated to capture the complexity of agent strategies in a repeated games environment (Neyman and Okada 1999). Complexity measures for gauging consumer demand and preferences have also been developed based on the quantity of information and configuration of information present in a given choice set (DeShazo and Fermo 2002). In a study that looked at the accuracy of market earnings forecasts, the portfolio complexity of research analysts was defined simply as the number of firms and industries being tracked in their market analysis (Clement 1999).

Measures of complexity have largely been applied at only a very superficial level in the social sciences, typically taking size as a simplistic basis for describing or capturing complexity. There are obvious deficiencies associated with size-based

complexity measures, the most evident being that not all large systems are complex (Edmonds 1999). It has been argued that the application of complex systems theory to the social sciences results in reductionist view of complexity when applied to the social domain (Lyon 1993). The important point raised is that contextual relationships such as political and moral issues are lost during the transformation into a mathematical or metaphoric model for complexity analysis.

7.2.2 Biological Sciences

It is especially difficult to define or measure biological complexity (Maynard Smith and Szathmary 1995; Szathmary, Jordan, and Pal 2001). An obvious measure would be the size of an organism's genome in terms of the number of base pairs (BP) present in the DNA, which can be thought of in the sense that a more complex organism would require lengthier instructions for making the organism (Cavalier-Smith 1985; Maynard Smith and Szathmary 1995). However, a total DNA count would place the complexity of humans (3.5×10^9 BP) an order of magnitude below a newt (19.0×10^9 BP) and two orders of magnitude below a lungfish (140.0×10^9 BP) and a lily (130.0×10^9 BP) (Maynard Smith and Szathmary 1995). An alternate measure of biological complexity based on DNA is that of counting only parts of the DNA that actually code for proteins that are expressed (Cavalier-Smith 1985). This complexity measure would then make more sense in that eukaryotes would have more coding DNA than prokaryotes, multi-celled organisms have more coding DNA than single-celled organisms, and that vertebrates have more coding DNA than invertebrates. However, this is an extremely coarse-grained classification that tells us very little about the structural and functional complexity between different organisms.

Another suggested measure of complexity for biological organisms based on genomic information is the number of genes present in the DNA (Szathmary, Jordan, and Pal 2001). However, humans, previously thought to have an order of magnitude more genes, are now estimated to have only around 20,000–35,000 genes and have the same order of magnitude of genes as the flowering plant *A. thaliana*

(25,498 genes), the nematode worm *C. elegans* (18,424 genes) and the fruit fly *D. melanogaster* (13,601 genes) (Szathmary, Jordan, and Pal 2001). As such, merely counting the number of genes as a measure of biological complexity may not be very insightful.

There are a number of other suggested methods of measuring biological complexity. Focusing on multicellular organism, the number of cell types present can be used to define the complexity of such an organism (Bonner 1988). A problem with this approach is that what constitute a distinct cell type as opposed to another cell type depends on our current understanding of molecular biology and biochemistry and may vary significantly between different groups of researchers (Nehaniv 2000a; Szathmary, Jordan, and Pal 2001).

The ability to measure the complexity of brains has been critically analyzed by Smith (1994). He laments

> "...how very far we are at present from being able to give a numerical estimate." (Smith 1994, p.93)

Showing the inadequacies of "borrowed" complexity measures from the physical sciences, he argues that *organization* as well as *levels of organization* need to be considered when attempting to capture the complexity of brains. A numerical measure based on the columnar organization of the neocortex was suggested citing a quantitative example that estimates the complexity of human brains, with roughly 300,000 such columns, to be 375 times greater than that of mouse brains, with only 800 columns.

A more formal approach based on algorithmic complexity to measuring biological complexity has been suggested by considering the number of developmental steps required to produce the organism from its DNA (Atlan and Koppel 1990). However, as critically pointed out by Szathmary, Jordan, and Pal (2001),

> "The snag here is that evolution is not an engineer but a tinkerer, so that there is no reason to expect that, for example, elephants have developed according to a minimalist program." (p.1315)

Another formal measure for biological complexity was suggested by Nehaniv (2000a) based on the notion of hierarchical complexity. In this measure, biological systems are assigned an integer value which gives the least number of hierarchical organized computing levels needed to construct an automata model of the biological system. Although powerful in terms of generalization since it does not require the actual knowledge of how a biological system is built nor its components, it does however have the requirement that the system can first be adequately modelled using finite automata (Nehaniv 2000a). The process of transforming biological systems into such automata is highly subjective and can be executed in a myriad of ways depending on how the system is viewed by the transformer. This measure of complexity was later applied to the measurement of evolvability in a later study and argued that open-ended evolutionary systems should show unbounded complexity increase over time (Nehaniv 2000b).

Szathmary, Jordan, and Pal (2001) more recently proposed the measurement of biological complexity by considering the connectivity of networks of transcription factors and the genes that regulate rather than direct counting of genes or the interactions among genes. They argue that biological complexity normally thought of in terms of morphological and behavioral complexity correlates better with the connectivity of gene-networks than direct measurements such as gene numbers since the former will correctly account for the presence of so-called *delegated information processing systems* in the form of vertebrate nervous and immune systems. However, current artificial evolutionary systems lack the level of sophistication in terms of such gene-regulatory networks and as such, do not readily lend themselves to such an analysis. Nonetheless, work has begun to imbue artificial evolutionary systems with some form of genetic regulation (Bongard 2002b) in the hope of evolving more sophisticated artificial organisms and will conceivably in the future allow for such a measure to be applied.

7.2.3 Physical Sciences

In the physical sciences literature, there are generally two widely-accepted views of measuring complexity. The first is an information-theoretic approach based on Shannon's entropy (Shannon 1948) and is commonly referred to as statistical complexity due to its formulation based on probability. Shannon's entropy measure $H(X)$ of a random variable X, where the outcomes x_i occur with probability p_i, is given by

$$H(X) = -\,C \sum_{i}^{N} p_i \log p_i \tag{7.1}$$

where C is the constant related to the base chosen to express the logarithm. It is a probabilistic measure of disorder present in a system and thus gives an indication of how much we do not know about a particular system's structure. Shannon's entropy is used to measure the amount of information content present within a given message or more generally any system of interest. Thus a more complex system would be expected to give a much higher information content than a less complex system. In other words, a more complex system would require more bits to describe compared to a less complex system. However, a sequence of random numbers will lead to the highest entropy and hence give a false indication of the system being complex when it is really just random. In this sense, complexity is somehow a measure of order or disorder that does not give a true indication of the information value present in the system, which in turn leads to an inaccurate characterization of complexity. Furthermore, an entropic measure does not take into account the semantic nature of the system. Consider for example a simple behavior such as walking. Let us assume that we are interested in measuring the complexity of walking in different environments and the walking itself is undertaken by an ANN. From Shannon's perspective, the complexity can be measured using the entropy of the data structure holding the neural network. Obviously a drawback for this view is its ignorance of the context and the concepts of embodiment and situatedness. The complexity of walking on a flat landscape is entirely different from walking on a rough landscape. Two neural networks may be represented using the same number of bits but exhibit entirely different behaviors. Using the outputs from the neural

networks as a measure of entropy is similarly problematic. Consider the case where a particular neural network optimized to perform robotic control has two of its output nodes swapped. The entropy as measured from the outputs of both the original and modified networks will remain the same but the behavior of the robot will change dramatically since the signals being sent to the individual actuators connected to these swapped output nodes have been disrupted. Hence, the change in the robot's behavior cannot be captured using this form of entropic measure.

The other approach to measuring complexity is a computation-theoretic approach based on Kolmogorov's application of universal Turing machines (Kolmogorov 1965) and is commonly known as Kolmogorov complexity or algorithmic complexity. It is a deterministic measure concerned with finding the shortest possible computer program or any abstract automaton that is capable of reproducing a given string. The Kolmogorov complexity $K(s)$ of a string s is given by

$$K(s) = \min\{|p| \mid s = C_T(p)\} \tag{7.2}$$

where $|p|$ represents the length of program p and $C_T(p)$ represents the result of running program p on Turing machine T. A more complex string would thus require a longer program while a simpler string would require a much shorter program. In essence, the complexity of a particular system is measured by the minimum amount of computation required to recreate the system in question. A well-known theoretical shortcoming of Kolmogorov complexity is that it is effectively incomputable since by virtue of the halting problem (Turing 1936), it cannot be determined with certainty that the absolute shortest program or description has been found (Badii and Politi 1997; Edmonds 1999; Shalizi 2001). On an empirical level, the following example will show the limitations of Kolmogorov complexity. Assume we have a sequence of random numbers. Obviously the shortest program which is able to reproduce this sequence is the sequence itself. Consequently, it is somehow also a measure of order or disorder, thereby endowing it with highly similar properties to that of Shannon's entropy (Badii and Politi 1997; Edmonds 1999). In addition, let us re-visit the neural network example. Assume that the robot is not using a fixed neural network but some form of evolvable hardware (which may be an evolutionary

neural network). If the fitness landscape for the problem at hand is monotonically increasing, a hill climber will simply be the shortest program which guarantees to re-produce the behavior. However, if the landscape is rugged, reproducing the behavior is only achievable if we know the seed. Otherwise, the problem will require complete enumeration to recreate the behavior. Unlike Shannon's entropy measure, Kolmogorov complexity is both a syntactic and semantic measure of complexity but ignores the pragmatic nature of the system. Furthermore, Kolmogorov complexity has been shown to be a poor measure for biological complexity (Smith 1994; Szathmary, Jordan, and Pal 2001).

A measure of complexity commonly discussed in computer science and software engineering literature is computational complexity, which is the time and storage space required by actual algorithms to solve a given problem (Badii and Politi 1997). Normally, it is referred to by the big-O notation which is a worst-case complexity measure that is defined as the order of the rate of growth of the resources required to compute the output to a problem as compared to the size of its input (Edmonds 1999). For example, an algorithm with computational complexity $O(n^2)$ would be expected to have a quadratic increase in computational resources with each linear increase in its input while an algorithm with computational complexity of $O(n^3)$ would be expected to have a cubic increase with each linear increase in its input. The analysis of computational complexity has important implications in the study of NP-completeness (Garey and Johnson 1979). The problem is to ascertain whether or not a particular problem is tractable or intractable, or more accurately to determine whether or not a polynomial time algorithm exists that can solve the problem on a von Neumann architecture. However, computational complexity provides only a rough approximation as it is measured only according to the order of the polynomial associated with the increase required in computational resources when there is an increase in input. Furthermore, this measure of complexity specifically looks at the construction of program code and how the computational cost is affected by this code. Again, it measures complexity at the syntactic level and thus is unable to accommodate notions of environments or interactions which would be

paramount in a study of embodied organisms.

Bennett (1988) proposed a measure of complexity called logical depth by combining the notions of Kolmogorov complexity and computational complexity. The logical depth $D_s(x)$ of a string s at level x is defined as

$$D_x(s) = \min\{T(p) \mid \ |p| - |p^*| \ < x \wedge U(p) = x\} \tag{7.3}$$

where p is the range of programs, $T(p)$ is the run-time required by program p, p^* is the smallest such program and U is a Turing machine. It essentially states that the logical depth of a string is based on the running time of the shortest algorithm that will reproduce a given string. It is poised between Kolmogorov complexity and computational complexity in that it considers the size of the shortest program as well as the run-time of the program respectively (Badii and Politi 1997). Logical depth was proposed as a measure of the value of information as reflected by the degree to which that information has been organized in a particular object. However, as logical depth is defined based on Kolmogorov complexity, it too is essentially incomputable (Badii and Politi 1997).

A starkly contrasting measure of complexity based on self-dissimilarity properties was recently proposed by Wolpert and MacReady (1997). Incorporating statistical inference and information theory, this complexity measure based on self-dissimilarity argues that the spatio-temporal signatures of complex systems vary markedly at different scales whereas the spatio-temporal signatures of simple systems do not differ significantly between different scales. Furthermore, the variation in a complex system's patterns over different space and time scales are considered to be the very essence of complexity rather than just an aberration of the modelling or measurement process. The spatio-temporal patterns in terms of the internal structure of a complex biological system for example, differs greatly when the observation scale is changed from the molecular level, to the cellular level, to the level of organs and to the level of the organisms itself. On the other hand, this self-dissimilarity measure argues that the spatio-temporal patterns of simple systems such as crystals, gases and even fractals do not change very much as one changes the scale of observation. However, as stated by the authors themselves, this notion of complexity has

only been formulated at the theoretical level and its real worth will only be proven when it is finally applied to real-world data (Wolpert and MacReady 1997).

In another research field known as computational mechanics, which is concerned with the dynamics of automata behavior, a mathematically-based entity that captures statistical complexity called the ϵ-machine was proposed by Feldman and Crutchfield (1998a). The ϵ-machine acts as a model for capturing the ensembles allowable configurations of a state machine. In other words, it is an object which allows for the inference of causal architecture from observed behavior (Shalizi 2001). As such, it allows for the definition and calculation of the global and macroscopic properties that reflect the average information processing capabilities of the system. The ϵ-machine has been applied empirically to measure the amount of self-organization achieved by four increasingly sophisticated types of process: memoryless transducers, time series, transducers with memory, and cellular automata (Shalizi 2001). Although the ϵ-machine has been shown to be effective and useful in capturing the increase in statistical complexity of such self-organized systems, this complexity measure is again based on automata theory and as such requires that the system being studied readily transforms into some form of state machine. As with other automata-based methods (Nehaniv 2000a), the question of how these transformations should be undertaken and what effects these transformations ultimately have when applied to less readily transformable systems such as creature behaviors and morphologies remains unanswered. More importantly, it provides a one dimensional view of complexity through the reduction of complex processes into a finite state machine and as such, does not leave any room for the interpretation of interactions between the system and its environment, for example. Hence, such automata-based complexity measures may not be a suitable methodology to apply to areas such as embodied cognition in terms of usefulness and pragmatic value where the essence of complexity lies in the system operating as a fully-interacting, adaptable and reactive whole.

7.3 Proposed EMO-Based Complexity Measure

We will now introduce the use of EMO as a convenient platform which researchers can utilize practically in attempting to define, measure, or simply characterize the complexity of everyday problems in a useful and purposeful manner. We first explain why a Pareto view to complexity is advantageous and then proceed to present our proposed method of measuring complexity using EMO. Finally, we discuss the assumptions associated with our proposed EMO-based complexity measure.

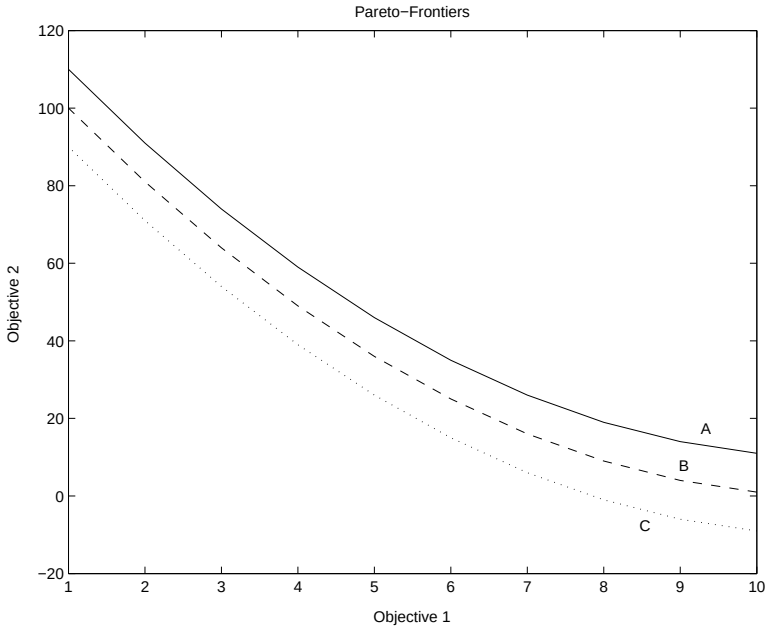


Figure 7.1: Diagram illustrating three different Pareto-frontiers for a problem with multiple objectives. X-axis: Objective 1, Y-axis: Objective 2.

Figure 7.1 provides an illustration of three layers of a potential Pareto-front of a particular multi-objective optimization problem. These layers can be viewed as providing three different levels of optimality. If we consider the problem to be maximization of both objectives, then the Pareto-front that dominates is curve A. All solutions along this front will dominate all other solutions in B and C because the solutions of A are more optimal along both objectives. If the problem involves minimization of both objectives, then the Pareto-front that dominates in this case

will be the curve C. We will show in the section that follows how this dominance ordering can provide useful insights for characterizing complexities from more than one perspective.

7.3.1 A Pareto View to Complexity

Here, we are interested in finding a binary relation which is able to say that one object "*is more complex than*" another object. There are a number of characteristics in which the elements in a set are related to one another and we shall visit each in turn to see which are desirable to have in such a binary relation. Assume a set A and a binary relation R on A. R is

- **Reflexive** if $(a, a) \in R \; \forall \; a \in A$. This is undesirable since an object should not be more complex than itself.
- **Irreflexive** if $(a, a) \notin R \; \forall \; a \in A$. This is desirable since an object should not be more complex than itself.
- **Symmetric** if $\forall \; a, b \in A, (a, b) \in R \rightarrow (b, a) \in R$. This is undesirable since if a is more complex than b, then b should not be more complex than a.
- **Asymmetric** if $\forall \; a, b \in A, (a, b) \in R \rightarrow (b, a) \notin R$. This is desirable since if a is more complex than b, then b should not be more complex than a.
- **Antisymmetric** if $\forall \; a, b \in A, (a, b) \in R$ and $(b, a) \in R \rightarrow a = b$. This is undesirable since if a and b are identical objects, then it should not hold true that a is more complex than b and b is more complex than a.
- **Transitive** if $\forall \; a, b, c \in A, (a, b) \in R$ and $(b, c) \in R \rightarrow (a, c) \in R$. This is desirable since if a is more complex than b, and b is more complex than c, then a should be more complex than c.
- **Negatively Transitive** if $\forall \; a, b, c \in A, (a, b) \notin R$ and $(b, c) \notin R \rightarrow (a, c) \notin R$. This is undesirable since if a is not more complex than b and b is not more complex than c, it does not imply that a is not more complex than c, which

we will show through contradiction. Assume two complexity measures 1 and 2 with three objects a, b, and c having the complexity values of (20,30), (30,10), and (10,20) respectively with reference to the complexity measures 1 and 2 in that order. In this case, a is not more complex than b since b has a higher value than a in terms of complexity measure 1. Similarly, b is not more complex than c since c has a higher value than b in terms of complexity measure 2. If the complexity relation R is negatively transitive, then this implies that a is not more complex than c. However, this is a contradiction as a is actually more complex than c since a has higher values in terms of both complexity measures. Therefore, this axiom is undesirable for the complexity binary relation R.

- **Connected** if $\forall\ a, b \in A, a \neq b \rightarrow (a, b) \in R$ or $(b, a) \in R$. This is undesirable since some pairs of objects may share the same complexity class and hence not all pairs of objects are necessarily connected through the relation that one object is more complex than the other. We will show that connectedness is an undesirable axiom using the example described above. a has higher values in terms of both complexity measures than c, hence a is more complex than c and therefore is connected to c through the complexity relation R. However, a is not more complex than b and thus shares the same complexity class as b, thus a is not connected to b through the complexity relation R. Similarly, b is not more complex than c and therefore b is also not connected to c. Thus, this axiom is undesirable for the complexity binary relation R since some objects may share the same complexity class.

- **Strongly Connected** if $\forall\ a, b \in A, (a, b) \in R$ or $(b, a) \in R$. This is undesirable since if the connectedness axiom does not hold true, then this axiom cannot hold true.

Therefore, the binary relation "is more complex than", R, should satisfy the irreflexivity, asymmetry and transitivity axioms.

It is important to point out that our purpose here is not to introduce another measure of complexity that can supposedly overcome all previous limitations

associated with existing measures neither claiming that it is an all-encompassing technique which will be able to calculate a definitive complexity value for complex systems. Our objective here is simply to propose and demonstrate that the Pareto set of solutions arising from an EMO process can be highly beneficial for characterizing and comparing between the complexities of different systems and at the same time satisfy the axioms desirable in a complexity binary relation. Furthermore, we will show through our experiments that the Pareto approach is a *useful* complexity measure. A complexity measure is said to be *useful* when it is able to capture what we intuitively regard as complex (Edmonds 1999).

There are two major advantages associated with using an EMO-based approach for capturing complexity. Firstly, it measures complexity of a particular system as seen from an observer's point of view. This has been argued by Casti (1986) to be paramount since the complexity of a system only has meaning through the interaction with its observer, particularly in more subjective areas such as behavioral complexity. As he puts it,

> "...system complexity is a contingent property arising out of the interaction I between a system S and an observer/decision-maker O. Thus, any perception and measure of complexity is necessarily a function of S, O, and I." (Casti 1986, p.149)

More importantly, he highlights the fact that

> "Conditioned by the physical sciences, we typically regard S as the active system, with O being a passive observer or disengaged controller. Such a picture misses the crucial point that generally the system S can also be regarded as an observer of O and that the interaction I is a two-way path." (Casti 1986, p.149)

Since a Pareto set is the result of optimization across two (or more) objectives, the solutions can be viewed as the result of a two-way interaction that occurs between the different objectives during the optimization process. Hence, a Pareto approach provides a distinct advantage when used to capture complexity by generating a set

of solutions that inherently exhibits properties of a two-way interaction and which can be reversibly used simply by looking at the results from the other optimization objective's view.

Secondly, we contend that the Pareto approach achieves a certain level of pragmatism when used as a complexity measure as opposed to simply providing a syntactic or semantic measure of complexity. In other words, it does not simply measure the complexity at the level of the language or symbols used to construct the system as in a typical syntactic measure nor does it measure the system's complexity within some predefined context or environment as would a semantic measure. Cariani (1992) explains that the syntactic axis represents operations conducted at the symbolic level, the semantic axis represents operations where symbolic information is extracted from the environment through measurement and control while the pragmatic axis represents the selection of appropriate measurements and controls that are advantageous to the operation of the system. In this sense, the proposed EMO methodology towards capturing complexity goes one step further in that it captures complexity through an evolutionary optimization process that continually generates new solutions from modification of previous solutions arising from testing and measurement of the system's performance within a given context or environment, which in turn is guided by the Pareto approach that imposes evolutionary pressures from multiple dimensions. In other words, it provides a view of complexity from a practical standpoint since a Pareto set comprises of solutions from a selection and adaptation process thereby constituting a pragmatic approach when such a Pareto set is used as a measure of complexity.

7.3.2 The Complexity Measure

We now present the formulation of our proposed complexity measure and demonstrate how it can be applied to characterize as well as compare the behavioral and morphological complexities of embodied artificial creatures. First, we define an embodied organism as the interaction between five components: morphology, behavior, controller, environment, and the learning algorithm. We will then show

how complexity can be defined as a partial order relation over this five-dimensional hyperspace. Accordingly, the complexity of two embodied organisms can be compared using this partial order relation. Finally, we support our argument with some experimental results which is presented in Section 7.4.

What follows is our proposal of a generic definition for complexity using the multi-objective paradigm. However, before we proceed with our definition, we need first to explain the concept of partial order.

Definition 1: Partial and Lexicographic Order. Assume the two sets A and B. Assume the l-subsets over A and B such that $A = \{a_1 < \cdots < a_l\}$ and $B = \{b_1 < \cdots < b_l\}$.

A partial order is defined as $A \leq_j B$ if $a_j \leq b_j,\ \forall j \in \{1, \ldots, l\}$

A Lexicographic order is defined as $A <_j B$ if $\exists a_k < b_k$ and $a_j = b_j,\ j < k,\ \forall j, k \in \{1, \ldots, l\}$

In other words, a lexicographic order is a total order. In multi-objective optimization, the concept of Pareto optimality is normally used. A solution x belongs to the Pareto set if there is not a solution y in the feasible solution set such that y dominates x (that is x has to be at least as good as y when measured on all objectives and better than y on at least one objective). The Pareto concept thus forms partial orders in the objective space.

Let us recall the embodied cognition problem. The problem is to study the relationship between the behavior, controller, environment, learning algorithm and morphology. A typical question that one may ask is "What is the optimal behavior for a given morphology, controller, learning algorithm and environment?". We can formally represent the problem of embodied cognition as the five sets B, C, E, L, and M for the five-dimensional hyperspace of behavior, controller, environment, learning algorithm, and morphology respectively. We also need to differentiate between the robot behavior B and the desired behavior $\hat{B}$. The former can be seen as the actual value of the fitness function and the latter can be seen as the real maximum of the fitness function. For example, if the desired behavior (task) is to maximize

the locomotion distance, then the global maximum of this function is the desired behavior, whereas the distance achieved by the robot (what the robot is actually doing) is the actual behavior. In traditional robotics, the problem can be seen as Given the desired behavior $\hat{B}$, find L which optimizes C subject to $E \bigcup M$. In psychology, the problem can be formulated as Given C, E, L and M, study the characteristics of the set B. In co-evolving morphology and mind, the problem is Given the desired behavior $\hat{B}$ and L, optimize C and M subject to E. A general observation is that the learning algorithm is usually fixed during the experiments.

In asking a question such as "Is a human more complex than a Monkey?", a natural question that follows would be "In what sense?". Complexity is not a unique concept. It is usually defined or measured within some context. For example, a human can be seen as more complex than a Monkey if we are looking at the complexity of intelligence, whereas a Monkey can be seen as more complex than the human if we are looking at the number of different gaits the monkey has for locomotion. Therefore, what is important from an artificial life perspective is to establish the complexity hierarchy on different scales. Consequently, we introduce the following definition for complexity.

Definition 2: Complexity is a strict partial order relation.

According to this definition, we can establish an order of complexity between the system's components/species. We can then compare the complexities of two species $S_1 = (B_1, C_1, E_1, L_1, M_1)$ and $S_2 = (B_2, C_2, E_2, L_2, M_2)$ as:

$$S_1 \text{ is at least as complex as } S_2 \textbf{ with respect to concept } \Psi \textbf{ iff}$$

$$S_2^{\Psi} = (B_2, C_2, E_2, L_2, M_2) \leq_j S_1^{\Psi} = (B_1, C_1, E_1, L_1, M_1),\ \forall j \in \{1, \ldots, l\},\ \textbf{Given}$$

$$B_i = \{B_{i1} < \cdots < B_{il}\},\ C_i = \{C_{i1} < \cdots < C_{il}\},\ E_i = \{E_{i1} < \cdots < E_{il}\},$$

$$L_i = \{L_{i1} < \cdots < L_{il}\},\ M_i = \{M_{i1} < \cdots < M_{il}\},\ i \in \{1, 2\} \tag{7.4}$$

where Ψ partitions the sets into l non-overlapping subsets.

We can even establish a complete order of complexity by using the lexicographic order as:

$$S_1 \text{ is more complex than } S_2 \textbf{ with respect to concept } \Psi \textbf{ iff}$$

$$S_2^{\Psi} = (B_2, C_2, E_2, L_2, M_2) <_j S_1^{\Psi} = (B_1, C_1, E_1, L_1, M_1),\ \forall j \in \{1, \ldots, l\},\ \textbf{Given}$$

$$B_i = \{B_{i1} < \cdots < B_{il}\},\ C_i = \{C_{i1} < \cdots < C_{il}\},\ E_i = \{E_{i1} < \cdots < E_{il}\},$$

$$L_i = \{L_{i1} < \cdots < L_{il}\},\ M_i = \{M_{i1} < \cdots < M_{il}\},\ i \in \{1, 2\} \tag{7.5}$$

The lexicographic order is not as flexible as partial order since the former requires a monotonic increase in complexity. The latter however, allows individuals to have similar levels of complexity. Therefore, it is more suitable for defining hierarchies of complexity. Hence, our definition of complexity based on the Pareto approach conforms to the set of axioms desirable in a binary operator for measuring complexity as discussed earlier in Section 7.3.1.

The concept of Pareto optimality is a special case of the partial order concept in that Pareto optimality is a strict partial order. In other words, Pareto optimality does not satisfy reflexivity; that is, a solution cannot dominate itself. Therefore two copies of the same solution cannot co-exist as Pareto solutions. Usually, when we have copies of one solution, we discard one of them. Therefore this problem does not arise when the Pareto set is generated. As a result, we can assume here that Pareto optimality imposes a complexity hierarchy on the solution set.

The previous definition will simply order the sets based on their complexities according to some concept Ψ. However, they do not provide an exact quantitative measure for complexity. In the simple case, given the five sets B, C, E, L, and M: assume the function f, which maps each element in each set to some value called the fitness, and assuming that C, E and L do not change, a simple measure of morphological change of complexity can be

$$\frac{\partial f(b)}{\partial m}, b \in B, m \in M \tag{7.6}$$

In other words, assuming that the environment, controller, and the learning algorithm are fixed, the change in morphological complexity can be measured in the

eyes of the change in the fitness of the robot (actual behavior). The fitness will be defined later in Section 7.4.2. Therefore, we introduce the following definition

Definition 3: Change of Complexity Value for the morphology is the rate of change in behavioral fitness when the morphology changes, given that both the environment, learning algorithm and controller are fixed.

The previous definition can be generalized to cover the controller and environment quite easily by simply replacing "morphology" by either "environment", "learning algorithm", or "controller". Based on this definition, if we can come up with a good measure for behavioral complexity, we can use this measure to quantify the change in complexity for morphology, controller, learning algorithm, or environment. In the same manner, if we have a complexity measure for the controller, we can use it to quantify the change of complexity in the other four parameters. Therefore, we propose the notion of defining the complexity of one object as viewed from the perspective of another object. This is not unlike Emmeche's idea of complexity as put in the eyes of the beholder (Emmeche 1994). However, we formalize and solidify this idea by putting it into practical and quantitative usage through the multi-objective approach. We will demonstrate that results from an EMO run of two conflicting objectives results in a Pareto-front that allows a comparison of the different aspects of an artificial creature's complexity.

In the literature, there are a number of related topics which can help here. For example, the Vapnik-Chervonenkis (VC) dimension (Vapnik and Chervonenkis 1971) can be used as a complexity measure for the controller. A feed-forward neural network using a threshold activation function has a VC dimension of $O(WlogW)$ while a similar network with a sigmoid activation has a VC dimension of $O(W^2)$, where W is the number of free parameters in the network (Haykin 1999). It is apparent from here that one can control the complexity of a network by minimizing the number of free parameters which can be done in a number of ways, the most obvious being the minimization of the number of synapses and/or the number of hidden units. It is important to separate between the learning algorithm and the model itself. For example, two identical neural networks with fixed architectures

may perform differently if one of them is trained using back-propagation while the other is trained using an evolutionary algorithm. In this case, the separation between the model and the algorithm helps us to isolate their individual effects and gain an understanding of their individual roles.

In this set of experiments, we are essentially posing two questions, what is the change of (1) behavioral complexity, and (2) morphological complexity of the artificial creature in the eyes of its controller. In other words, how complex is the behavior and morphology in terms of evolving a successful controller?

7.3.3 Complexity Measures Revisited

Before we proceed with an empirical experiment of how this complexity measure based on the Pareto concept can be applied to capturing the morphological and behavioral complexities of artificially evolved creatures, we first provide some examples of how this methodology can be applied in a more general manner to the biological, social and physical sciences.

First, we provide a Pareto view to complexity in the biological sciences. More specifically, we will use two existing measures for biological complexity, namely genome length and number of genes which were discussed previously in Section 7.2.2. The data used in this example are actual genomic information extracted from the **EnsEMBL** on-line database (EnsEMBL.Org 2002)[2]. We will compare the complexities of five different organisms (the version of the organism's genomic database is given in following parentheses): human (v.8.30a.1), mouse (v.8.3b.1), zebrafish (v.8.08.1), fugu or pufferfish (v.8.1.1) and mosquito (v.8.1b.1). In terms of genome length, the order of complexity from least to greatest number of DNA base-pairs, we obtain

1. zebrafish (0.04×10^9 BP)

[2]**EnsEMBL** is a joint project between European Molecular Biology Laboratory, European Bioinformatics Institute and the Sanger Institute to develop a software system that produces and maintains automatic annotation on eukaryotic genomes. It is one of the three main repositories for genomic information (Gibas and Jambeck 2001).

2. mosquito (0.28×10^9 BP)

3. fugu (0.33×10^9 BP)

4. mouse (2.73×10^9 BP)

5. human (3.34×10^9 BP)

However, if we take the number of genes instead as the measure for biological complexity, then we obtain the following ordering

1. zebrafish (1511 genes)

2. mosquito (15088 genes)

3. mouse (22444 genes)

4. human (22980 genes)

5. fugu (31059 genes)

As such, by simply changing the complexity measure (scale) from genome length to number of genes, we have dramatically changed the ordering of complexity for the mouse, human and fugu, as depicted in Figure 7.2.

Now let us take a multi-objective approach to characterizing the complexities of these different organisms by combining the two biological complexity measures into a 2D graph.

Let us assume that a real biological organism can be made analogous to an artificial embodied creature. Now we can compare between the organisms' complexities by making the following representations: assume that all the organisms share a common environment E being the Earth, acquire knowledge through some common learning mechanism L such as reinforcement learning, that the organisms have different morphologies M, that the genome is acting as a master controller C and that the primal behavior in the organism B is a reflection of its genes.

Using the Pareto approach, we can now characterize the complexity of these five organisms at four different levels. Firstly, we can make some observations

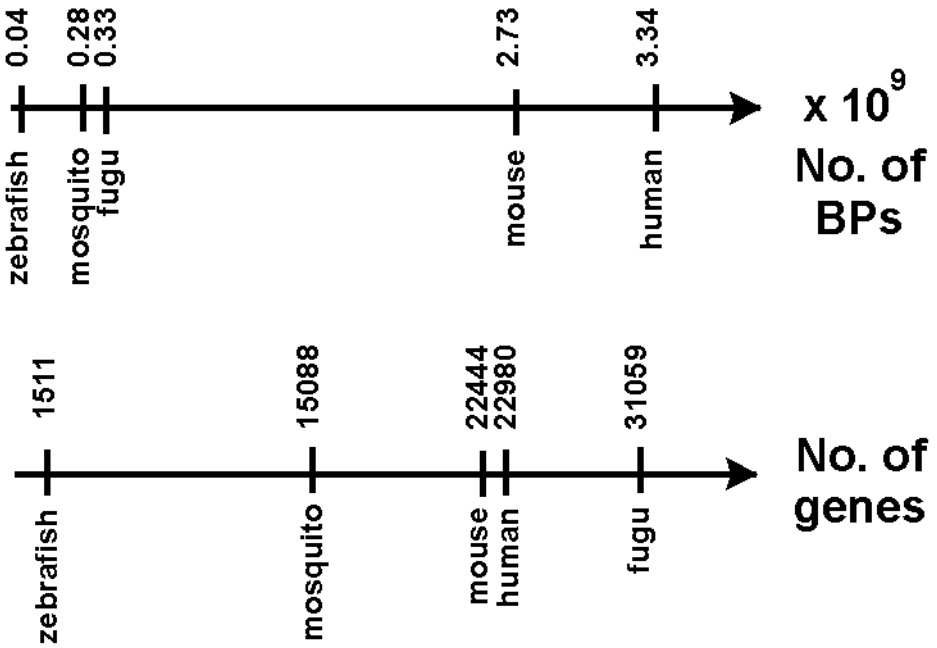


Figure 7.2: Conflicting ordering of biological complexity when taking a single-objective view to complexity measures (scales).

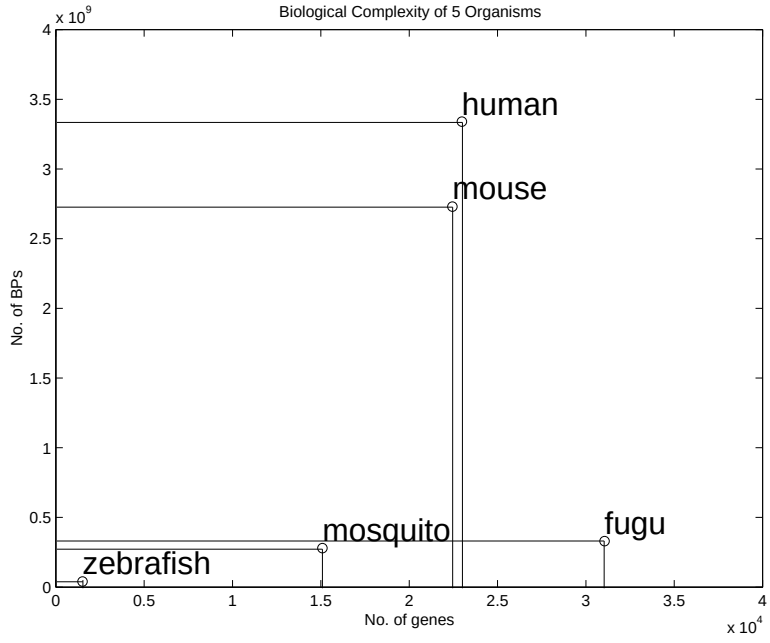


Figure 7.3: Comparing the biological complexity of 5 organisms by combining two different complexity measures (scales). X-axis: No. of genes, Y-axis: No. of BPs.

about which organisms are more complex than others by considering the dominance ordering present in one set of organisms defined within the two complexity dimensions B and C. Since the mosquito dominates the zebrafish along both complexity dimensions, we can say that the mosquito is more complex than the zebrafish. In a similar manner, we can also say that the mouse is more complex than the mosquito and that both the human and fugu is in turn more complex than the mouse. However, if we compare the human against the fugu, we cannot say that either is more complex than the other because they both each dominate the other along one of the complexity dimensions. In other words, the human and fugu are at the Pareto-front of this particular set of organisms when compared using these two measures of biological complexity. In this case, the hierarchy of complexity can be illustrated as in Figure 7.4.

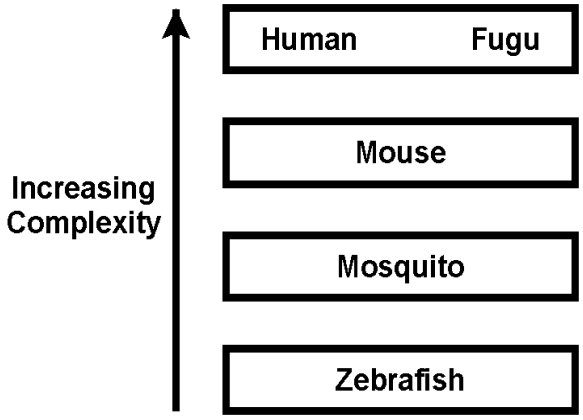


Figure 7.4: Diagram illustrating the complexity hierarchy of 5 organisms constructed using a Pareto approach.

Secondly, we can make some quantitative comparisons by looking at the Euclidean distance between these organisms. We can see that the increase in complexity from the zebrafish to the mosquito was smaller than the increase in complexity from the mosquito to the mouse. We can also look at the change in complexity along one dimension relative to the other. For example, a relatively small change in the zebrafish's number of DNA base-pairs resulted in a surprisingly large increase in number of genes as compared to the required increase in number of DNA base-pairs

going from the mosquito to the mouse for a roughly similar increase in the number of genes.

Thirdly, if we had another dimension of complexity, say in terms of the environment E and that E now represents the planet Mars, then this second set of organisms from Mars can be compared with the first set of organisms from Earth by taking the two Pareto-fronts present in these two sets of organisms and finding some common ground for complexity along one dimension and comparing the relative change in the other. In this case, let us presume that Organism M-human and Organism M-fugu represent the two organisms placed at the Pareto-front of organisms from Mars, then these two organisms can be compared against the human and fugu from Earth, since these form the Pareto-front of organisms from Earth. For example, say if we find that Organism M-human has a comparable number of DNA base-pairs as the fugu but has double the number of genes, then we might conclude that the change in the environment from Earth to Mars shows a more complex environment in Mars in terms of the required number of genes to survive as compared to on Earth since for the same number of DNA base-pairs, the organism in Mars required more number of genes to allow for the survival of Organism M-fugu.

In terms of the social sciences, an example can be taken from the political governance of two different countries, which can be regarded as two distinct complex systems. Let us assume that Country A has less citizens than Country B and define the complexity of the country based on the number of citizens that needs to be governed. Using this measure, Country A would be considered less complex than Country B. Let us assume that another measure of complexity can be formulated in terms of the governance structure of the different countries. Assume Country A is democratic with a government that is lead by a group of elected representatives while Country B is autocratic with a government that is lead by a single dictator. It is reasonable to assume that Country A requires many interactions between its politicians before any decision can be made compared with no interaction required whatsoever in the case of a decision made by the sole dictator in Country B. In this sense, Country A may now be considered to be more complex than Country B on

this new scale. A Pareto view will again help in comparing between the political complexities of these countries by considering one complexity measure in the eyes of the other complexity measure rather than taking these conflicting viewpoints as stand-alone indicators, which would then reflect a failure in providing a reasonable characterization of political complexity between the two countries by virtue of the contrasting quantizations.

Let us now see how this can be done in terms of our formulated measure of complexity. First, let us consider that the actual political complexity of these countries are the result of interactions between both its citizens and governance structure, rather than just either of these singular components. Then let us assume that the political complexities of the countries represent the morphology M of the complex system, that the environment E is unchanged if we consider them to be geographically located in the same region of the world, that learning occurs through some common medium L such as the mass media, that the populations represent the behavior B and that the governance structures represent the controller C. Now we can compare the change in political complexity between these two different countries ∂M by measuring some quantitative change in the behavior of the population ∂B through some commonality that can be found in the hyperspace of the controlling governance structure C. Conversely, we can compare the governance complexity between the two countries ∂M from the reverse viewpoint by measuring some quantitative change in terms of the controlling governance structure ∂C through establishing some commonality in the hyperspace of the population's behavior B. Casti (1986) provides an elegant example of how such a complex system emerges from the interactions between its governance structure and its citizens. Here, he states that the citizens views the governance structure as complex if the actions taken by its political leaders seem to be incomprehensible:

> "...they [*the citizens*] see a byzantine and unwieldy government bureaucracy and a large number of independent decision-makers (government agencies) affecting their day-to-day life." (p.150)

Similarly, the government typically also views its citizens as being very complex:

> "They [*the political leaders*] would see a seemingly fickle, capricious public, composed of a large number of independent self-interest groups clamoring for more and more public goods and services." (p.150)

Hence, we can see that a multi-objective view to the study of social sciences such as political complexity can again be very insightful and valuable.

In revisiting complexity measures for the physical sciences, let us turn to an example from computer science itself. Wuensche (1999) has recently devised a method for automatic classification of 1D cellular automata (CA) rules into one of three dynamical groups, that is ordered, complex and chaotic systems based on the frequency of particular updating rules being looked-up over time called the input-entropy. Based on Shannon's entropy measure, Wuensche (1999) formulated input-entropy S at time-step t as

$$S^t = -\sum_{i=1}^{2^k}(\frac{Q_i^t}{n} \times log(\frac{Q_i^t}{n})) \tag{7.7}$$

where k and n are the neighborhood and system size of the CA, and Q_i^t is the lookup frequency of neighborhood i at time t. One of the proposed classification methods was based on indications given by two measures: (1) the input-entropy itself, and (2) the variability (standard deviation) of the input-entropy. An example from Wuensche (1999) classified three rules from a Boolean CA system with $k = 5$ and $n = 150$, described as typical examples of CA behaviors, as given in Table 7.2:

Rule No.	Classification	Input-Entropy	Variability of Input-Entropy
01 dc 96 10	Ordered	low	low
6c 1e 53 a8	Complex	medium	high
99 4a 6a 65	Chaotic	high	low

Table 7.2: Classification of 3 cellular automata rules according to Wuensche (1999).

Let us now consider two rules arising from the same CA setup that after experimentation and analysis gave the following indications:

- Rule X: moderately high entropy, moderately high variability

- Rule Y: very high entropy, very high variability

It would be difficult to classify these two rules since they are placed mid-way between the existing classes. However, if we take a multi-objective view to this problem, we would be able to provide the following perspective:

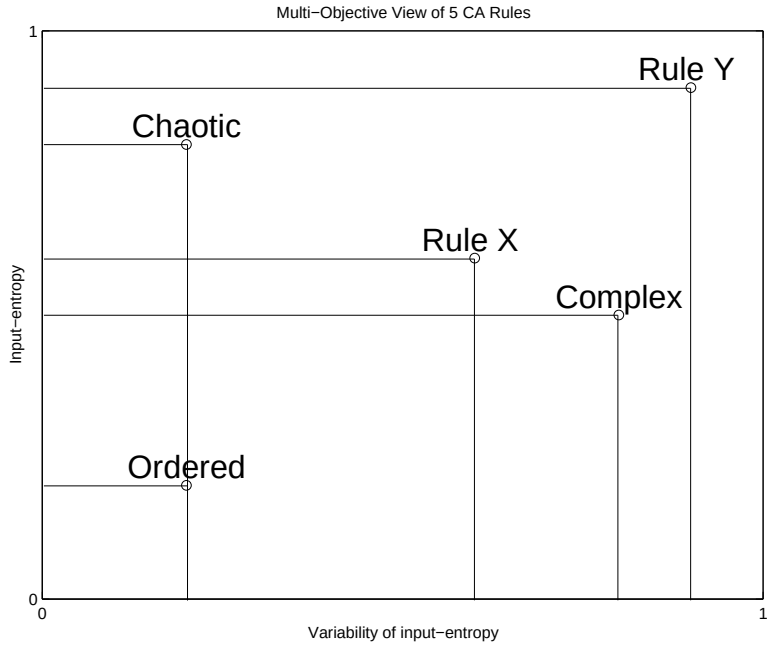


Figure 7.5: Multi-objective view of 5 cellular automata rules by combining input-entropy and variability of input-entropy. X-axis: Variability of input-entropy, Y-axis: Input-entropy.

If we now cast the cellular automata system into our proposed measure of complexity, we can formulate the following representations. The environment E is represented by the CA's overall system setup as defined by the neighborhood size, periodic boundary conditions and dimensionality. As such, E remains constant since both rules arise from the same CA setup. L would be null in all cases since no learning occurs in a CA system. Let us now consider that the difference between Rule X and Rule Y represents a change in the controller C of the CA system since the dynamics of a CA is dependent on the rule being used in the CA. Next, we shall consider that the morphology M of the CA is represented by the input-entropy and that the behavior B of the CA is represented by the variability in the input-entropy.

Firstly, we can see that Rule Y dominates all other rules in this particular system since it has higher values for both complexity dimensions of B and M. As such, we can say that Rule Y is the most complex rule from a multi-objective perspective. Also, we can see that a Pareto-front is formed by the Chaotic and Complex rules, and that both rules are as complex as each other since they both dominate each other in one dimension. Furthermore, since Rule X is not dominated by either the Chaotic nor the Complex rule in both dimensions, it too belongs to the Pareto set for this particular CA system and that its complexity can be characterized as being similar to that of the Chaotic and Complex rules in terms of B and M. As for the Ordered rule, it has the same value for B (variability of input-entropy) as the Chaotic rule and has a lower value for M (input-entropy) than the Chaotic rule. As such, the Ordered rule is dominated by the Pareto-front of which the Chaotic rule is a member. Hence, it can be characterized as being the least complex among all the rules in this particular CA system since it is dominated by all other rules. As with the biological example visited earlier, a change in the environment E, for example increasing the neighborhood size, will produce a second set of observations which can then be compared with this first set of observations by comparing the two Pareto-fronts obtained from these two different CA systems.

In the next section, we will describe the setup of the experiments which demonstrate empirically how our proposed measure for capturing complexity can be applied to the comparison of the morphological and behavioral complexities of artificially evolved creatures.

7.4 Experimental Setup

7.4.1 Two Artificial Creatures

Two artificial creatures were used in this study (Figure 7.6): (1) a quadruped creature with four legs, (2) and a hexapod creature with six legs. The first artificial creature (Figure 7.6.1) is the same quadruped used in Chapters 4, 5, and

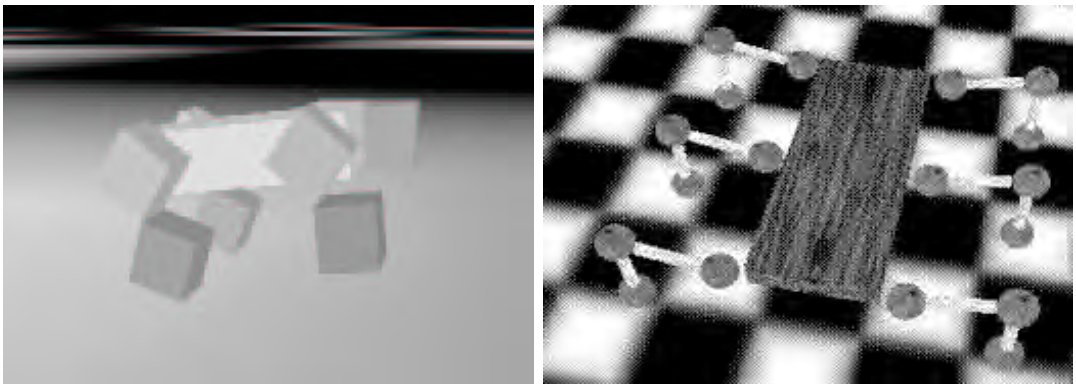


Figure 7.6: Screen dump of the 1. quadruped (left), 2. hexapod (right) artificial creatures.

6. The second artificial creature (Figure 7.6.2) is a hexapod[3] with 6 long legs that are connected to the torso by insect hip joints. Each insect hip joint consists of two hinges, making it a joint with two degrees of freedom: one to control the back-and-forth swinging and another for the lifting of the leg. Each leg has an upper limb connected to a lower limb by a hinge (one degree of freedom) joint. The hinges are actuated by motors in the same fashion as in the first artificial creature.

Morphological Characteristic	Simulated Quadruped	Simulated Hexapod
No. of legs	4	6
Degrees of freedom	8	24
No. of sensors	12	24
No. of motors	8	18

Table 7.3: A comparison of the simulated quadruped and hexapod creatures' morphological characteristics.

Table 7.3 presents a comparison of the main features of the two artificial creatures. It would appear that the quadruped has a much simpler design compared to the hexapod creature. However, this is only a subjective observation from a human designer's perspective. It remains to be seen whether this view will hold when

[3]The design and experimentation of the hexapod creature was carried out jointly with another graduate student Ms. Minh Ha Nguyen.

we compare the complexities of these two artificial creatures from the controller's and behavior's perspectives.

7.4.2 Controller Architecture

The Pareto-frontier of our evolutionary runs are obtained from optimizing two conflicting objectives as in Chapter 5: (1) minimizing the number of hidden units used in the ANN that act as the creature's controller, and (2) maximizing horizontal locomotion distance of the artificial creature. What we obtain at the end of the runs are again Pareto sets of ANNs that trade-off between number of hidden units and locomotion distance. The locomotion distances achieved by the different Pareto solutions will provide a common ground where locomotion competency can be used to compare different behaviors and morphologies. It will provide a set of ANNs with the smallest hidden layer capable of achieving a variety of locomotion competencies. The structural definition of the evolved ANNs can now be used as a measure of complexity for the different creature behaviors and morphologies.

The type of ANN architecture used for the experiments in this chapter is NNType3 as presented in Section 3.3.3, which has fully-connected feed-forward network with recurrent connections on the hidden units as well as direct input-output connections. Only one type of architecture was used since the results from Chapter 5 showed no significant differences between the four architectures. Of the four architectures, NNType3 was chosen since the best overall locomotion distance was achieved using this particular architecture. A diagrammatic representation of part of the ANN architecture is illustrated in Figure 3.4.4. The genotype representation used for specifying the ANN controller remains unchanged as explained in Section 3.4 and the SPANN algorithm as presented in Section 5.4.1 was again used to drive the artificial evolutionary process.

7.4.3 Assumptions

Two assumptions need to be made. First, the Pareto set obtained from evolution is considered to be the actual Pareto set. This means that for the creature

on the Pareto set, the maximum amount of locomotion is achieved with the minimum number of hidden units in the ANN. We do note however that the evolved Pareto set in the experiments may not have converged to the optimal set. Nevertheless, it is not the objective of this paper to provide a method which guarantees convergence of EMO but rather to introduce and demonstrate the application of measuring complexity in the eyes of the beholder. It is important to mention that although this assumption may not hold, the results can still be valid. This will be the case when creatures are not on the actual Pareto-front but the distances between them on the intermediate Pareto-front are similar to that of creatures on the actual Pareto-front.

The second assumption that we are making is that there are no redundancies present in the ANN architectures of the evolved Pareto set. This simply means that all the input and output units as well as the synaptic connections between layers of the network are actually involved in and required for achieving the observed locomotion competency. We have investigated the amount of redundancy present in evolved ANN controllers in Section 6.4.5 and found that the self-adaptive Pareto EMO approach produces networks with virtually no redundancy.

Before excluding these assumptions, it is important to emphasize that none of these assumptions will dramatically change our findings. Since we are interested in the partial order and the rate of change, getting to the exact Pareto-front or obtaining the neural network with zero redundancy may not affect the results. Take for example the solutions on the Pareto set found during a hypothetical EMO run $X_1 = (0, 5), X_2 = (1, 4)$. Let us assume that the two solutions on the actual Pareto-front are $Y_1 = (0, 5.5), Y_2 = (1, 4.5)$. What we are interested in is the difference between the two solutions and the partial order between them, which as can be seen in this case, are not affected by the evolved Pareto-front not being the actual Pareto-front. Obviously this is a hypothetical example but at least it demonstrates that the assumptions may hold in a large number of actual cases.

7.4.4 Evolutionary Runs

Two series of experiments were conducted. Behavioral complexity was investigated in the first series of experiments and morphological complexity was investigated in the second. For both series of experiments, each evolutionary run was allowed to evolve over 1000 generations with a randomly initialized population size of 30. The maximum number of hidden units was again fixed at 15 as in previous experiments carried out in Chapters 4, 5, and 6. The number of hidden units used and maximum locomotion achieved for each genotype evaluated as well as the non-dominated set of solutions obtained in every generation were recorded. The Pareto solutions obtained at the completion of the evolutionary process were compared to obtain a characterization of the behavioral and morphological complexity.

To investigate behavioral complexity in the eyes of the controller, the morphology was fixed by using only the quadruped creature but the desired behavior was varied by having two different fitness functions. The first fitness function measured only the maximum horizontal locomotion achieved but the second fitness function measured both maximum horizontal locomotion and static stability achieved. By static stability, we mean that the creature achieves a statically stable locomotion gait with at least three of its supporting legs touching the ground during each step of its movement. The two problems we have are:

$(P1)$

$$f_1 = \Uparrow \ d(g) \tag{7.8}$$

$$f_2 = \Downarrow \ \sum_{h=0}^{H} \rho_h \tag{7.9}$$

$(P2)$

$$f_1 = \Uparrow \ d(g)/20 + s(g)/500 \tag{7.10}$$

$$f_2 = \Downarrow \ \sum_{h=0}^{H} \rho_h \tag{7.11}$$

where $P1$ and $P2$ are the two sets of objectives used. f_1 and f_2 represent the respective fitness functions used to evaluate the genotypes g. d refers to the locomotion distance achieved and s is the number of times the creature is statically stable as

controlled by the ANN at the end of the evaluation period of 500 timesteps. $P1$ is using the locomotion distance as the first objective while $P2$ is using a linear combination of the locomotion distance and static stability. Minimizing the number of hidden units is the second objective in both problems.

To investigate morphological complexity, another set of 10 independent runs was carried out but this time using the hexapod creature. This is to enable a comparison with the quadruped creature which has a significantly different morphology in terms of its basic design. The $P1$ set of objectives was used to keep the behavior fixed. The results obtained in this second series of experiments were then compared against the results obtained from the first series of experiments where the quadruped creature was used with the $P1$ set of objective functions. Where the $P1$ experiments involving the quadruped creature was required, the results from Section 5.6 for NNType3 were used since the setup of the experiments were identical.

7.5 Results and Discussion

7.5.1 Morphological Complexity

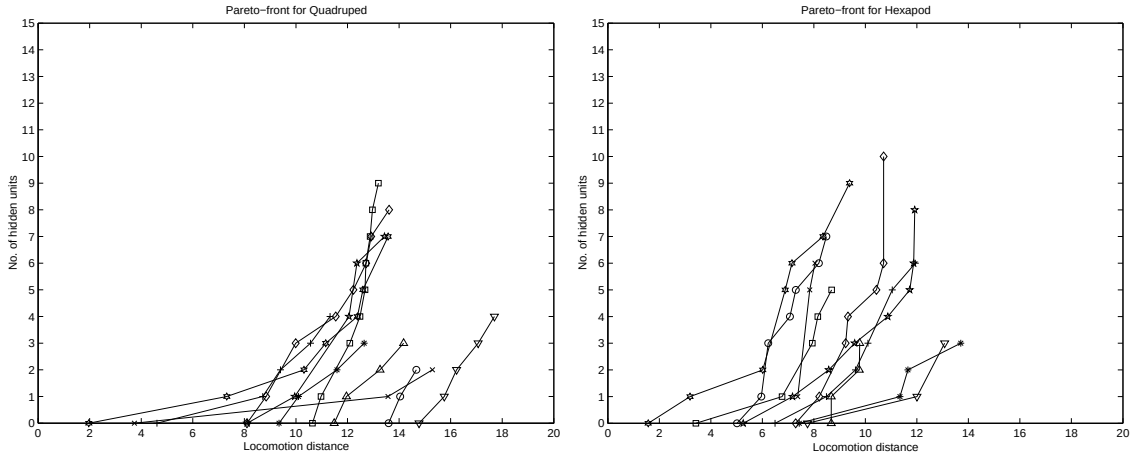


Figure 7.7: Pareto-frontiers of controllers obtained from 10 runs using the $P1$ set of objectives for the 1. quadruped (left), 2. hexapod (right). X-axis: Locomotion distance, Y-axis: No. of hidden units.

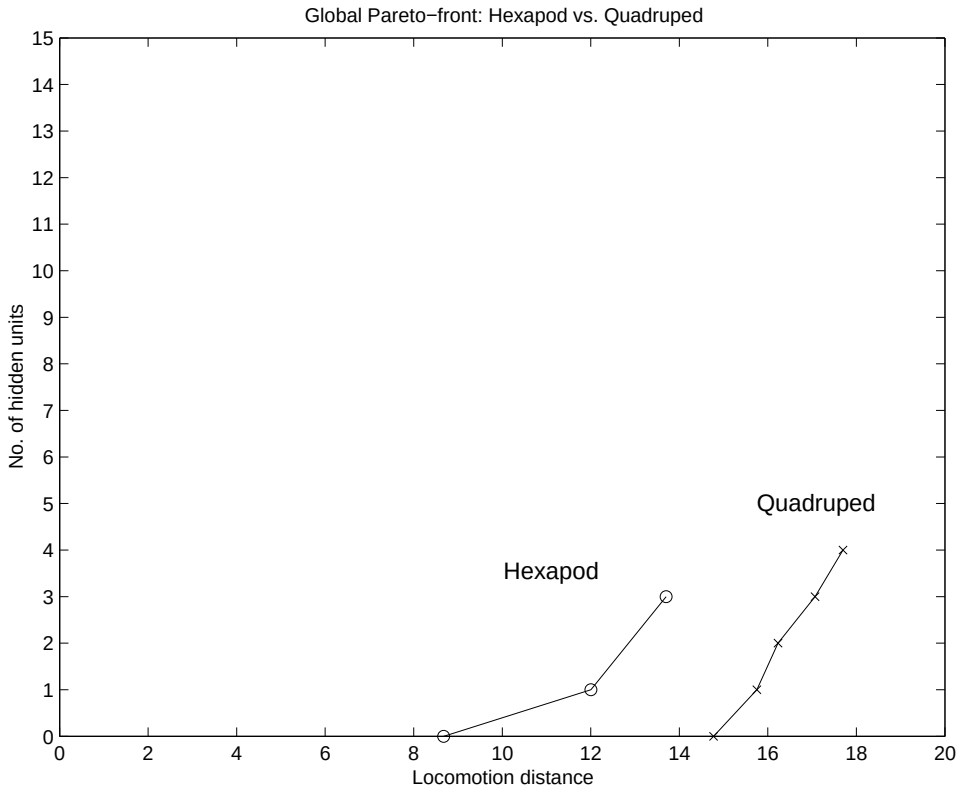


Figure 7.8: Global Pareto-front of controllers obtained from 10 runs using the $P1$ set of objectives for the quadruped and hexapod. X-axis: Locomotion distance, Y-axis: No. of hidden units.

We first present the results for the quadruped and hexapod evolved under $P1$. Figure 7.7 compares the Pareto optimal solutions obtained for the two different morphologies over 10 runs. Figure 7.8 plots the global Pareto-front for both the hexapod and quadruped. As such, we are comparing two Pareto-fronts that characterize the complexities of two different systems. Here we are fixing E and L. Therefore, we can either measure the change of morphological complexity in the eyes of the behavior or the controller: that is, $\frac{\partial f(B)}{\partial M}$ or $\frac{\partial f(C)}{\partial M}$ respectively. If we fix the actual behavior B as the locomotion competency of achieving a movement of $13 < d < 15$, then the change in the controller $\partial f(C)$ is measured according to the number of hidden units used in the ANN. At this point of comparison, we find that the quadruped is able to achieve the desired behavior with 0 hidden units whereas the hexapod required 3 hidden units (Figure 7.8). Therefore, this is an indication

that from the controller's point of view, given the change in morphology ∂M from the quadruped to the hexapod, there was an increase in complexity for the controller ∂C from 0 hidden units to 3 hidden units. Hence, the hexapod morphology can be seen as being more complex than the quadruped morphology in the eyes of the controller.

Conversely, if we would like to measure the complexity of the morphology from the eyes of the locomotion behavior, we can choose a common point of comparison in terms of the network size. If we fix the controller C to having a hidden layer size of 3 hidden units, then the change in the locomotion behavior $\partial f(B)$ is measured according to the maximum distance achieved by artificial creatures. At this point of comparison, we find that the quadruped achieves just over 17 units distance while the hexapod is only able to achieve just under 14 units distance (Figure 7.8). Thus, this is an indication that from the locomotion behavior's point of view, given the change in morphology ∂M from the quadruped to the hexapod, there was an increase in complexity for the locomotion behavior ∂B of approximately 3 units distance. In this case, the quadruped morphology can be seen as being more complex than the hexapod morphology.

As such, by taking different viewpoints, we find different interpretations of which morphology is more complex than the other. This is not unlike what we have seen in the biological sciences (see Section 7.2.2) where different complexity measures result in different orderings of organismic complexity. Therefore, a Pareto approach to capturing complexity is advantageous in the sense that it gives multiple views of complexity through a single comparison exercise.

7.5.2 Behavioral Complexity

A comparison of the results obtained using the two different sets of fitness functions $P1$ and $P2$ is presented in Table 7.4. Here we are fixing M, L and E and looking for the change in behavioral complexity. The morphology M is fixed by using the quadruped creature only. For $P1$, we can see that the Pareto-frontier offers a number of different behaviors. In this case, we are comparing complexities within a

Type of Behavior	Pareto Controller	No. of Hidden Units	Locomotion Distance	Static Stability
$P1$	1	0	14.7730	19
	2	1	15.7506	24
	3	2	16.2295	30
	4	3	17.0663	26
	5	4	17.6994	14
$P2$	1	0	5.2065	304
	2	1	3.3355	408
	3	2	3.5935	420
	4	3	3.6829	419

Table 7.4: Comparison of number of hidden units, locomotion distance and static stability for global Pareto optimal controllers obtained using the quadruped for the $P1$ and $P2$ sets of objective functions.

system by using the evolved Pareto-front to represent the complexity characteristics of a single system. For example, a network with no hidden units can achieve up to 14.7 units of distance while the creature driven by a network with 5 hidden units can achieve 17.7 units of distance within the 500 timesteps. This is an indication that to achieve a higher speed gait entails a more complex behavior than a lower speed gait.

We can also see the effect of static stability, which requires a walking behavior, by comparing the two Pareto-fronts that characterize the $P1$ and $P2$ systems respectively. By comparing a running behavior using a dynamic gait in $P1$ with no hidden units against a walking behavior using a static gait in $P2$ with no hidden units, we can see that using the same number of hidden units, the creature achieves both a dynamic as well as a quasi-static gait. If more static stability is required, this will necessitate an increase in controller complexity.

At this point of comparison, we find that the behavior achieved with the $P1$ fitness functions consistently produced a higher locomotion distance than the behavior achieved with the $P2$ fitness functions. This meant that it was much harder for the $P2$ behavior to achieve the same level of locomotion competency in terms of distance moved as the $P1$ behavior due to the added sub-objective of having

to achieve static stability during locomotion. Thus, the complexity of achieving the $P2$ behavior can be seen as being at a higher level of the complexity hierarchy than the $P1$ fitness function in the eyes of the controller.

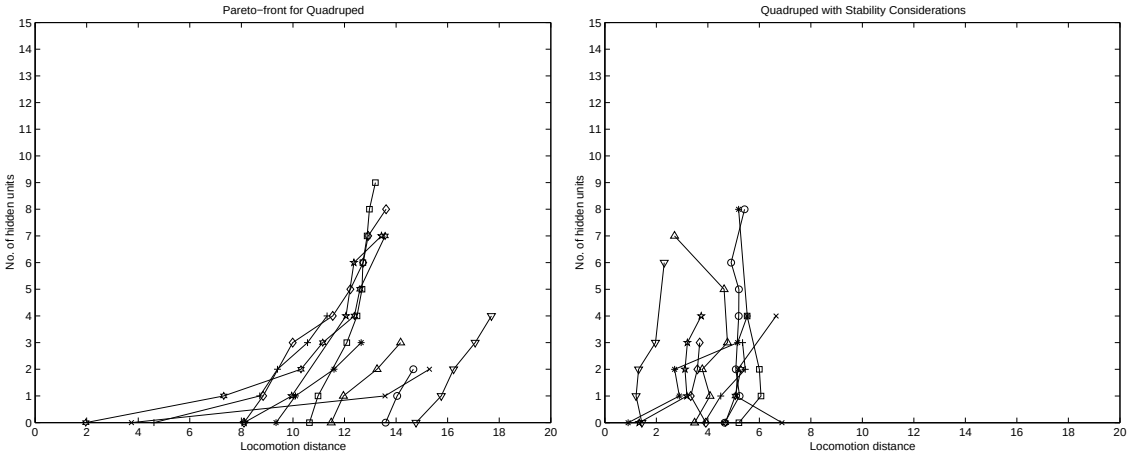


Figure 7.9: Locomotion distance of Pareto solutions obtained from 10 runs using the quadruped with the $P1$ (left) and $P2$ (right) sets of objective functions. X-axis: Locomotion distance, Y-axis: No. of hidden units.

Figure 7.9 depicts the locomotion distance achieved using the quadruped with the $P2$ set of objectives, which measure for both locomotion distance and static stability, along with the Pareto-fronts obtained from using the $P1$ set of objectives. Note that the graph for $P2$ does not depict Pareto-fronts since we are only interested in the locomotion distance, which is only part of the objective function. Here we see that the locomotion distance achieved was much lower due to the added sub-objective of attempting to maximize static stability. This is expected since the creature will be discouraged from jumping behaviors, which may allow for greater locomotion capabilities. No controllers could be found that achieved a locomotion distance of $d > 7$. As such, it was not possible to compare the behavioral complexities of standard locomotion versus locomotion with stability since the quadruped evolved for standard locomotion achieved a distance of $13 < d < 15$ with the least complex network of 0 hidden units. In other words, no commonality could be found that would have enabled a comparison of the change in behavioral complexity from

the eyes of the controller.

7.5.3 Limitations

From our experiments, we note the following limitations with the proposed complexity measure based on the Pareto approach:

- To compare across Pareto-fronts from different systems, some common ground needs to be established to enable a fair comparison of complexity. The first disadvantage arising from this requirement is that the inherent complexity of the particular component being investigated may be so different between the different systems that no common ground exists. For example, we found that no common ground existed when we tried to compare between the two fitness functions $P1$ and $P2$ when viewed from the controllers perspective (see Section 7.5.2). Secondly, the actual determination of the range of values considered to be within this area of commonality is dependent on the results of the components being compared. For example, in our comparison between the quadruped and hexapod creatures, we determined that the locomotion competency range used as the common ground when viewing from the controller's perspective was $13 < d < 15$ by virtue of the results that were obtained (see Section 7.5.1).

- In empirical studies involving artificial evolutionary systems, some form of conflicting objectives needs to be present before this Pareto approach will provide any additional benefits in taking a multi-objective view to complexity. In the case where the objectives of a particular problem are not in conflict, the Pareto approach will simply reflect an hierarchical ordering similar to those obtained when the individual single objectives are used to characterize the objects in question. As such, this requirement of having conflicting objectives is not so much of a disadvantage but rather a desirable characteristic to have for the Pareto approach to be able to provide a different view towards capturing complexity. For example, if the evolutionary runs conducted in our experiments

had maximization of locomotion distance and straight-line walking behavior, then there exists no conflict since the maximum distance will be achieved when the walking behavior results in a path that is maximally straight. In other words, this complexity measure will not provide any additional value towards characterizing complexity unless the system in question can be formulated in some form of conflicting components.

7.6 Chapter Summary

In the introductory sections of this chapter, we have reviewed the concept of complexity as well as a number of existing measures of complexity as applied in the social, biological and physical sciences. We then proposed a Pareto approach towards complexity and revisited each of these areas to show how conflicting measures of complexity can be re-formulated using a Pareto approach to provide an understanding of complexity from different perspectives. Subsequently, we proceeded to demonstrate how this technique can be applied empirically for studying the behavioral and morphological complexities of artificially evolved embodied creatures. In doing so, we found that the morphological complexity of a quadruped creature was lower than the morphological complexity of a hexapod creature as seen from the perspective of an evolving locomotion controller. At the same time, the quadruped was found to be more complex than the hexapod in terms of behavioral complexity. This proposed measure will allow for artificial creatures with evolvable morphologies to be compared in terms of their morphological as well as behavioral complexity in the next chapter.

Chapter 8

Co-Evolution of Morphology and Mind

Morphology and materials are intimately related to control in adaptive behavior (Pfeifer 2000). This is referred to as ecological balance and was argued to enable better designs of robots and other artificial organisms. There is a trade-off between morphology and control: having the right morphology can greatly simplify controller requirements. As such, it was also argued that discussions of embodied autonomous agents pertaining only to neural processing issues are incomplete without a related discussion of the agent's shape, physical properties of its sensors and motors as well as the materials which make up the agent's body and appendages.

The term co-evolution as used in the field of evolutionary computation usually refers to the concurrent evolution of two or more populations with fitness functions that are coupled dynamically (Rosin and Belew 1997). These co-evolutionary algorithms can either consist of competing populations (Hillis 1992; Angeline and Pollack 1993; Cliff and Miller 1996; Rosin and Belew 1997; Nolfi and Floreano 2000; Floreano, Nolfi, and Mondada 2001) or cooperative populations (Husbands and Mill 1991; Paredis 1995; Potter and De Jong 2000). However, here we use the term co-evolution as previously used by Dellaert and Beer (1994), Lee, Hallam, and Lund (1996), and Hornby and Pollack (2001a) to refer to the simultaneous evolution of both morphology and controller in evolving artificial creatures rather than to refer

to competing or cooperating populations with coupled fitness functions.

In this chapter, we will attempt to evolve both the creature's morphology and mind. This will be achieved by relaxing certain morphological constraints imposed on the artificial creature where the focus thus far has only been on the optimization of its ANN controller. We will now include the parameters of the creature's ANN controller as well as morphology into the chromosome to allow for both aspects of the creature's body and mind to be optimized simultaneously through co-evolution. As we have seen from the literature survey conducted in Chapter 2, there are two extremes in evolving artificial creatures where on the one hand all of the morphology is totally fixed and unchangeable such as in the more common four-legged and six-legged physical robots, and on the other hand virtually all aspects of the morphology are changeable such as in abstract robots. Thus, we have adopted an approach midway between these extremes and evolve only certain parameters of the artificial creature's morphology and maintain some underlying body plan representative of quadrupedal organisms. The objective of this chapter is thus to explore not only the legged locomotion behavior that can be achieved by co-evolving both the morphology and controller but also the selective adaptation of body parts and joint constraints by evolution for efficient legged locomotion.

8.1 Additional Chromosome Parameters

To allow for the morphology of the creature to be simultaneously co-evolved with its ANN controller, we relax two specific aspects of the quadruped's morphology. The first constraint relaxed is the length of each of the upper and lower limbs in each of the four legs, which are now variable in length. The second constraint relaxed is the manner in which each limb is connected to the adjoining body part, where a choice of how the upper limb joins to the torso as well as how each lower limb joins to each upper limb is now available. These additional evolutionary parameters arising from the relaxation of the morphological constraints are added to the existing chromosome structure as previously explained in Section 3.4. More ex-

planation concerning the addition of these new genes into the chromosome is given in the following paragraphs.

Limb Description	Previous Length	New Length
Upper back left	1.0	$0.2 + (L1 \times 4.0)$
Upper front left	1.0	$0.2 + (L2 \times 4.0)$
Upper back right	1.0	$0.2 + (L3 \times 4.0)$
Upper front right	1.0	$0.2 + (L4 \times 4.0)$
Lower back left	1.0	$0.2 + (L5 \times 4.0)$
Lower front left	1.0	$0.2 + (L6 \times 4.0)$
Lower back right	1.0	$0.2 + (L7 \times 4.0)$
Lower front right	1.0	$0.2 + (L8 \times 4.0)$

Table 8.1: Description of the simulated quadruped's previous fixed limb lengths and new evolvable limb lengths (in centimeters).

Table 8.1 lists the artificial creature's previous and new limb lengths. Previously, the length of each limb was fixed to only 1cm. In the new design, each limb has a minimum length of 0.2cm and to that is added a length that can vary from 0 to 4cm. These variable attributes are denoted *L1* through *L8* and are included into the chromosome as continuous variables taking values between 0 and 1. As such, the dimensions of the limbs are now $1 \times 1 \times (0.2 + (L \times 4))$cm.

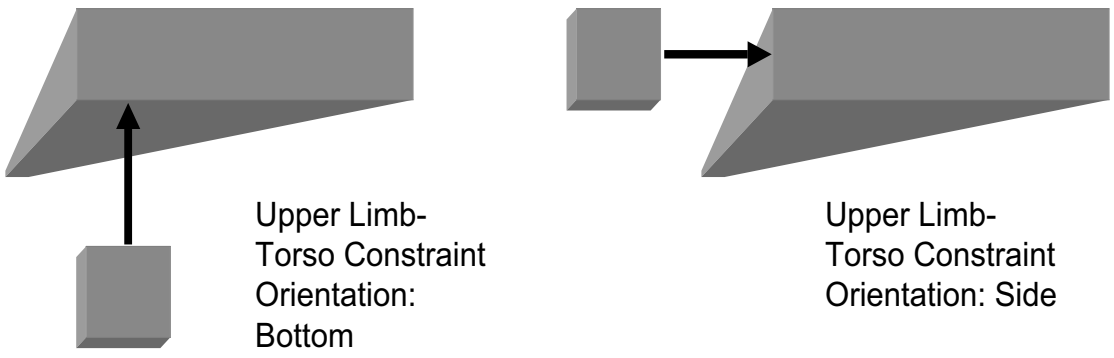


Figure 8.1: Front-on view of the evolvable constraint orientation for the joint connection between the torso and upper limbs 1. bottom-oriented connection (left), 2. side-oriented connection (right).

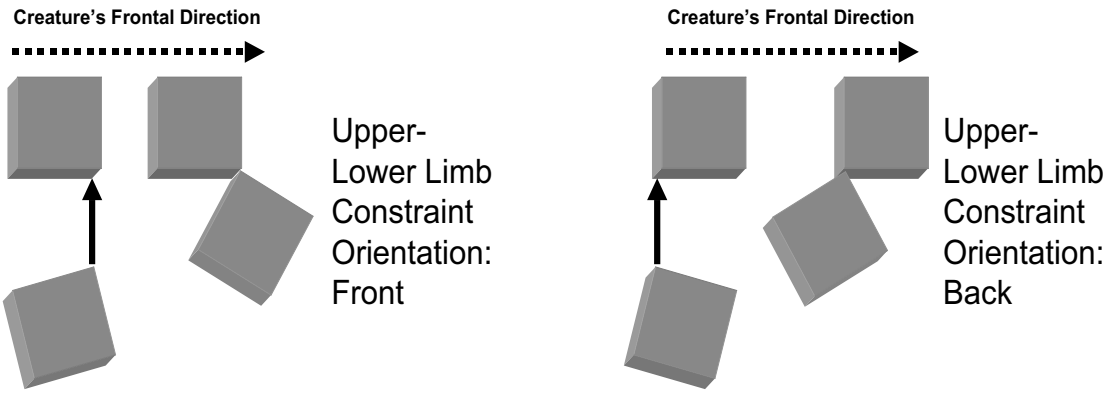


Figure 8.2: Side-on view of the evolvable constraint orientation for the joint connection between upper and lower limbs 1. front-oriented connection (left), 2. back-oriented connection (right).

Limb Description	**Connected To (Limb)**	**Previous Constraint Orientation**	**New Constraint Orientation**
Upper back left	Torso	Side	Side or Bottom (*C1*)
Upper front left	Torso	Side	Side or Bottom (*C2*)
Upper back right	Torso	Side	Side or Bottom (*C3*)
Upper front right	Torso	Side	Side or Bottom (*C4*)
Lower back left	Upper back left	Back	Back or Front (*C5*)
Lower front left	Upper front left	Back	Back or Front (*C6*)
Lower back right	Upper back right	Back	Back or Front (*C7*)
Lower front right	Upper front right	Back	Back or Front (*C8*)

Table 8.2: Description of the simulated quadruped's previous fixed constraint orientations and new evolvable constraint orientations.

Table 8.2 lists the artificial creature's previous and new limb constraint orientations. In the previous setup, all joint connections were fixed to either side or back orientations depending on whether it was an upper or lower limb. In the new setup, there is now a choice of constraint orientation for each of the eight joint connections. Each upper limb can now be connected to the torso either from side (Figure 8.1.1) or from the bottom (Figure 8.1.2). Each lower limb can now be connected to each upper limb either from the front (Figure 8.2.1) or from the back (Figure 8.2.2). These variable attributes are denoted *C1* through *C8* and are

included into the chromosome as Boolean variables that denote either one of the two orientation choices.

8.2 Experimental Setup

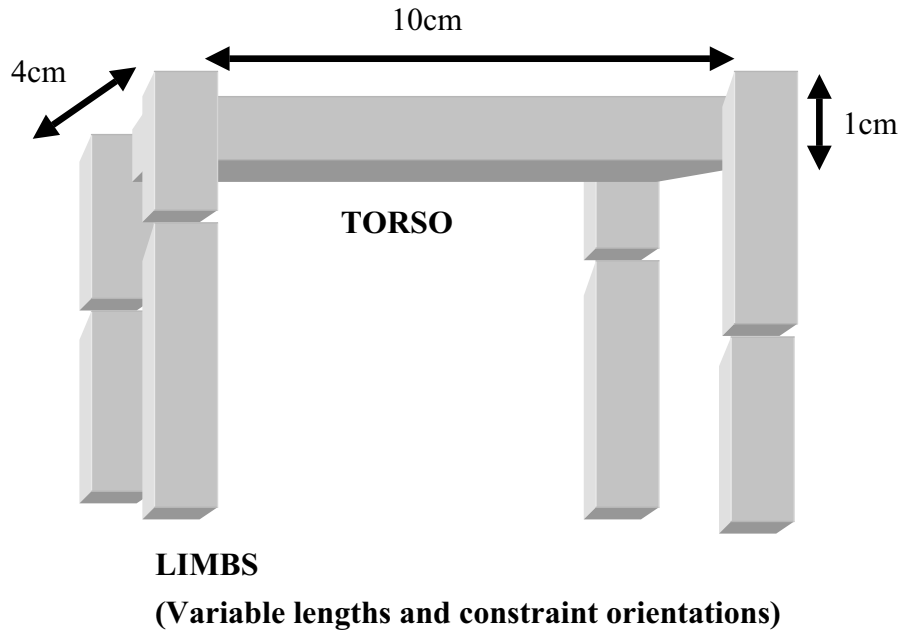


Figure 8.3: A geometric description of the new artificial creature used in the co-evolutionary experiments with evolvable limb lengths and evolvable constraint orientations. The torso dimensions are fixed as denoted.

A series of 10 independent runs were carried out to investigate the co-evolution of morphology and controller simultaneously. SPANN was used as the algorithm for driving the artificial evolutionary optimization process again. This augmented version which allows for the co-evolution of morphology and mind is denoted as the SPANN-CMM algorithm. A number of the artificial creature's setup was changed in SPANN-CMM to accommodate the additional evolvable components of its morphology. All changes were scaled linearly in relation to the original quadruped's setup discussed in Section 3.2. A geometric description of the new

artificial creature is given in Figure 8.3. The torso dimension was increased to $10 \times 4 \times 1$cm and the mass increased to 5g. From initial co-evolutionary experiments, the original shorter torso length of 4cm caused the limbs with longer lengths to constantly come in contact with each other during the locomotion cycle and thus restricted full movement of the limbs. Hence the length of the torso was increased to 10cm to allow for full and unhindered movement of the longer limbs. Furthermore, the creature frequently toppled over when longer limbs were present, therefore the width of the torso was also increased from 2cm to 4cm to allow for greater stability. As outlined in the introduction to this chapter, we wished to maintain some basic quadrupedal body plan and as such, we kept the torso fixed as a hand-designed component of the artificial creature and not part of the evolvable morphology. The mass of each limb and force generated at each associated hinge joint were also scaled linearly in accordance with the length of the associated limbs as they evolved. The maximum rotation allowed at each hinge joint remained unchanged at 1.57 radians. The ANN architecture used was NNType3 since it gave the best overall results from prior experiments. All other evolutionary and simulation parameters remained the same: 1000 generations, 30 individuals, 500 timesteps and a maximum of 15 hidden units allowed in the ANN controller.

In the analysis, we first discuss the best solutions obtained from the evolutionary runs using SPANN-CMM in terms of the locomotion behavior and size of the ANN controller as well as comparing the results against SPANN. This is followed by a simple characterization of the search space associated with this co-evolutionary setup. We then use the complexity measure defined in the previous chapter to conduct a simple comparison of the different creatures evolved with and without co-evolution of morphology.

8.3 Results and Discussion

Table 8.3 shows the best results obtained using SPANN-CMM in terms of locomotion distance. The overall best f_1 fitness was slightly better than SPANN

Algorithm	Overall Best f_1 Fitness	Worst of Best f_1 Fitness	Average Best f_1 Fitness ± Standard Deviation	t-statistic (against SPANN)
SPANN-CMM	18.1472	11.9002	15.1421 ± 2.0321	1.63
SPANN	17.6994	11.3234	13.9626 ± 1.7033	-

Table 8.3: Comparison of best locomotion distance for Pareto solutions found using the SPANN-CMM and SPANN algorithms over 10 independent runs.

achieving a locomotion distance of 18.1 units. The average locomotion distance of the best evolved controllers were also slightly higher than those obtained using SPANN although the improvement was not statistically significant.

Table 8.4 shows the best results obtained using SPANN-CMM in terms of number of hidden units used in the ANN controllers. The overall best f_2 fitness was slightly worse than SPANN using 1 more hidden unit. The worst of the best f_2 fitness was also slightly worse than SPANN using 2 more hidden units. The average number of hidden units used in the best evolved controllers was also slightly higher than those obtained using SPANN although the increase was not statistically significant.

Algorithm	Overall Best f_2 Fitness	Worst of Best f_2 Fitness	Average Best f_2 Fitness ± Standard Deviation	t-statistic (against SPANN)
SPANN-CMM	3	11	6.4 ± 2.8	1.15
SPANN	2	9	4.9 ± 2.6	-

Table 8.4: Comparison of smallest hidden layer size for Pareto solutions found using the SPANN-CMM and SPANN algorithms over 10 independent runs.

In general, the results in terms of the best solutions obtained at the end of co-evolving both the morphology and controller were not very different for both the locomotion distances achieved as well as number of hidden units used in the ANN controller.

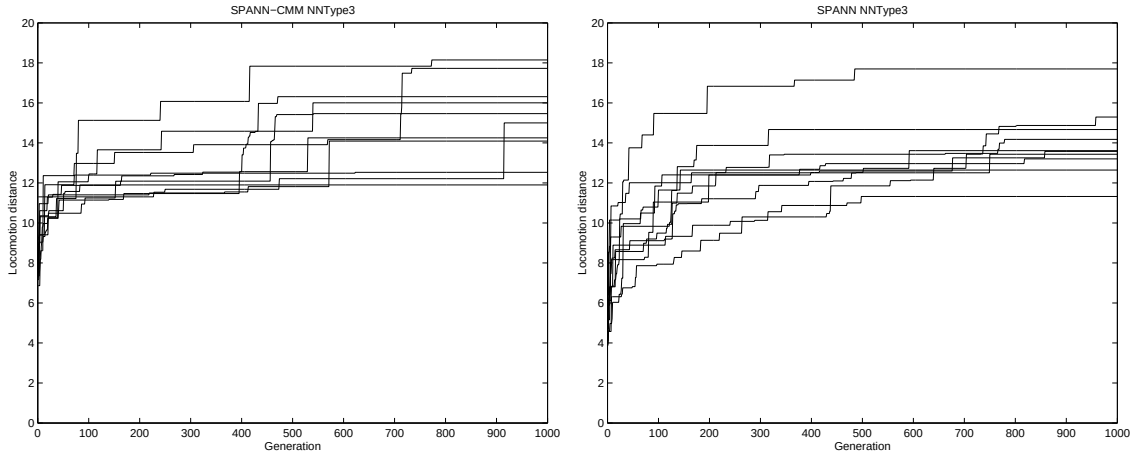


Figure 8.4: Comparison of best locomotion distance of Pareto solutions obtained over 1000 generations for 10 runs using the 1. SPANN-CMM (left), 2. SPANN (right) algorithms. X-axis: Generation, Y-axis: Locomotion distance.

8.3.1 Evolutionary Dynamics

Two interesting characteristics emerged in the evolutionary dynamics of the best solutions in SPANN-CMM (Figure 8.4.1). Firstly, although some significant improvements in the locomotion fitness were observed early during the evolutionary process, significant improvements still occurred during the later stages of evolution. In six of the runs, large improvements occurred between the 400–600th generation. One of these six runs later showed another large improvement around the 700th generation. Another separate run showed a large improvement as late as the 900th generation. Secondly, the improvements were very discrete in nature, some improving over 3 units of locomotion distance in a single generation. Both these characteristics were in contrast to those observed in SPANN where the majority of the improvements occurred well before the 200th generation and occurred much more gradually (Figure 8.4.2). This suggests firstly that evolution might have discovered a significantly better morphology during the co-evolutionary optimization process and hence a large improvement in locomotion distance could be achieved within a single generation. Secondly, SPANN-CMM may require more time to converge on a solution by virtue of the increased number of evolutionary parameters

resulting from the inclusion of additional morphological parameters into the chromosome. Thus, the solutions found may still be some distance away from the actual Pareto-frontier of optimal solutions. In order to test this second postulation, we extended the best run (the tenth seed), which still showed a noticeable improvement at the 773rd generation, beyond 1000 generations for another 500 generations from the 773rd generation but found no further improvements. Hence, the second postulation that the co-evolutionary runs require more time to converge is unlikely to be true since the best run had in fact converged to a final solution by the 1000th generation.

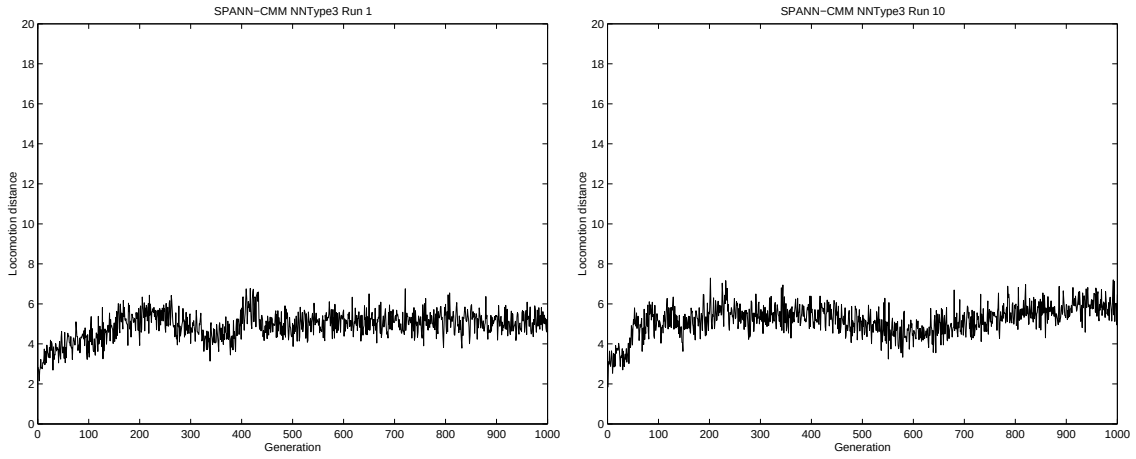


Figure 8.5: Mean locomotion distance of population over 1000 generations for the SPANN-CMM algorithm using the 1. first (left), 2. tenth (right) seeds. X-axis: Generation, Y-Axis: Locomotion distance. Additional graphs can be found in the accompanying CD-ROM.

The variation in SPANN-CMM's population mean tended to either remain fixed within a certain range or increase slightly over time, as illustrated by Figures 8.5.1 and 8.5.2 respectively. The standard deviation in the population was generally quite high, varying mostly between 3 and 4 in the earlier half of evolution then tending towards 5 in the later half of evolution, as shown by Figure 8.6.1, and even 6 in some of the runs, as shown by Figure 8.6.2. This is most probably due to the generation of new individuals which have morphologies that may not be easily

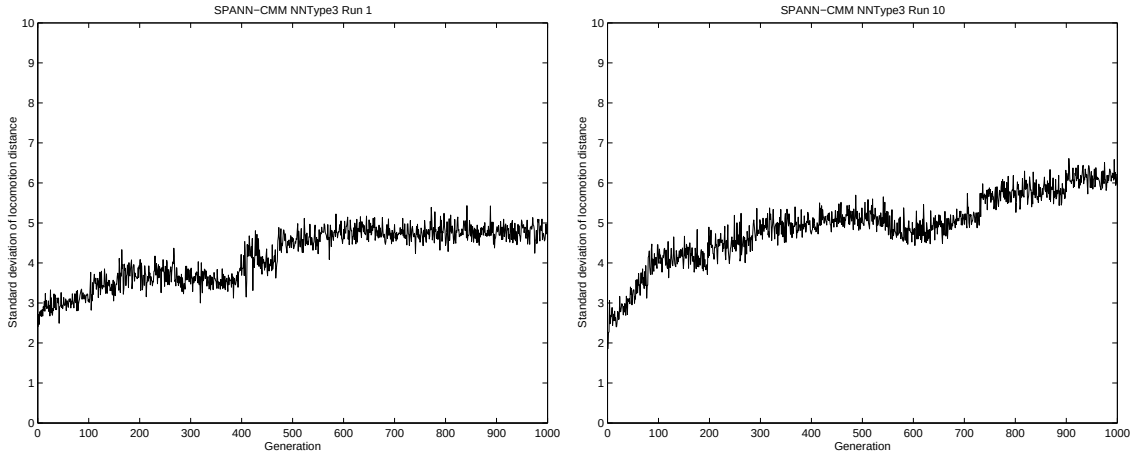


Figure 8.6: Standard deviation for locomotion distance of population over 1000 generations for the SPANN-CMM algorithm using the 1. first (left), 2. tenth (right) seeds. X-axis: Generation, Y-Axis: Standard deviation of locomotion distance. Additional graphs can be found in the accompanying CD-ROM.

controlled by the existing ANN controllers. Consequently, a larger gap will exist between the optimized solutions that have controllers and morphologies that work well together and the newly generated solutions that don't as evolution progresses.

8.3.2 Comparing Pareto-Fronts and Morphological Complexity

Table 8.5 lists the global Pareto solutions obtained using SPANN-CMM followed by those obtained using SPANN. The Pareto-front is highly similar to that obtained using SPANN although SPANN-CMM did have one more solution on the Pareto-front which used 6 hidden units and achieved the best locomotion distance. In general, the locomotion distances achieved with each hidden layer size was highly similar. No trend could be observed with the controllers that were comparable, with three hidden layer sizes achieving slightly higher locomotion distances in SPANN-CMM (networks using 1, 2 & 3 hidden units) and two hidden layer sizes performing slightly better in SPANN (networks using 0 & 4 hidden units).

Figure 8.7 compares the Pareto optimal solutions obtained using SPANN-

Algorithm	No. of Hidden Units	Locomotion Distance
SPANN-CMM	0	14.2775
	1	15.8432
	2	17.0130
	3	17.2338
	4	17.6614
	6	18.1472
SPANN	0	14.7730
	1	15.7506
	2	16.2295
	3	17.0663
	4	17.6994

Table 8.5: Comparison of number of hidden units and locomotion distance for global Pareto optimal controllers obtained using the SPANN-CMM and SPANN algorithms over 10 independent runs.

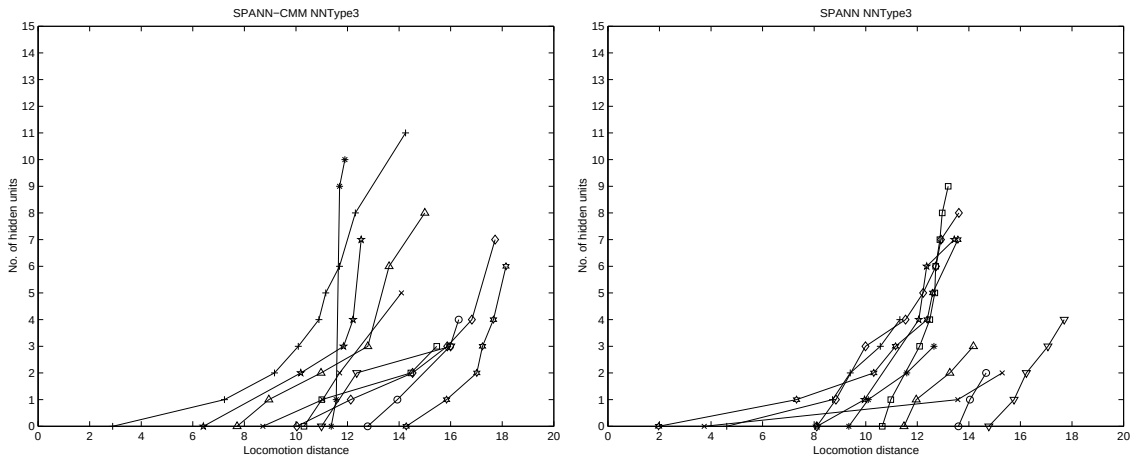


Figure 8.7: Pareto-front of solutions obtained for 10 runs using the 1. SPANN-CMM (left), 2. SPANN (right) algorithms. X-axis: Locomotion distance, Y-axis: No. of hidden units.

CMM against SPANN over 10 runs. As in Section 7.5.1, we are comparing two Pareto-fronts that characterize the complexities of two different morphologies. Here the environment E and learning algorithm L are again fixed, so we can either measure the change of morphological complexity in the eyes of the behavior or the controller: that is, $\frac{\partial f(B)}{\partial M}$ or $\frac{\partial f(C)}{\partial M}$ respectively. If we fix the actual behavior B as the locomotion competency of achieving a movement of $13 < d < 15$, then the change

in the controller $\partial f(C)$ is measured according to the number of hidden units used in the ANN. At this point of comparison, we find that both SPANN-CMM (Figure 8.7.1) and SPANN (Figure 8.7.2) produced creatures that were able to achieve the desired behavior with 0 hidden units. Therefore, this is an indication that from the controller's point of view, given the change in morphology ∂M from the creature evolved by co-evolving morphology and controller to the fixed morphology creature, there was no increase in complexity for the controller ∂C. Hence, the SPANN-CMM morphology can be seen as being at the same level of complexity as the SPANN morphology in the eyes of the controller.

Conversely, we can also measure the complexity of the morphology from the eyes of the locomotion behavior. First we need to choose a common point of comparison in terms of the network size. If we fix the controller C to having a hidden layer size of 3 hidden units, then the change in the locomotion behavior $\partial f(B)$ is measured according to the maximum distance achieved by artificial creatures. At this point of comparison, we find again that the creatures evolved with both SPANN-CMM (Figure 8.7.1) and SPANN (Figure 8.7.2) achieve a similar locomotion distance of 17 units. Thus, this is an indication that from the locomotion behavior's point of view, given the change in morphology ∂M from the co-evolved morphology and fixed morphology, there was no increase in complexity for the locomotion behavior ∂B. In this case, the SPANN-CMM morphology can again be seen as having the same level of complexity as the SPANN morphology.

Table 8.6 lists the evolved values for the variable parameters in the creature's morphology for the global Pareto solutions found by SPANN-CMM. The limb length values for Pareto solutions 2 through 5 were very similar while solutions 1 and 6 were slightly more different. In Pareto solutions 2 through 5, the only difference in limb length was found in gene *L5* of 0.1cm. Pareto solution 1 had a longer limb length for genes *L5* and *L6* while Pareto solution 6 had a longer limb length for gene *L4* compared to the other Pareto solutions. There was more variation in terms of the constraint orientation found in the Pareto solutions although some common choices of orientation could still be found. All *C1*, *C5* and *C8* genes had similar

Pareto Solution No.	1	2	3	4	5	6
Locomotion Distance	14.3	15.8	17.0	17.2	17.7	18.1
No. of Hidden Units	0	1	2	3	4	6
L1	0.2	0.2	0.2	0.2	0.2	0.2
L2	0.2	0.2	0.2	0.2	0.2	0.2
L3	4.1	4.1	4.1	4.1	4.1	4.1
L4	0.2	0.2	0.2	0.2	0.2	3.8
L5	4.0	2.7	2.8	2.7	2.8	2.8
L6	3.8	0.2	0.2	0.2	0.2	0.2
L7	0.6	0.6	0.6	0.6	0.6	0.2
L8	0.2	0.2	0.2	0.2	0.2	0.2
C1	side	side	side	side	side	side
C2	bottom	bottom	bottom	side	bottom	side
C3	side	side	side	side	side	bottom
C4	side	bottom	bottom	side	side	side
C5	back	back	back	back	back	back
C6	back	back	back	back	front	back
C7	back	front	front	back	front	back
C8	front	front	front	front	front	front

Table 8.6: Evolved limb lengths and constraint orientations for global Pareto optimal controllers obtained using the SPANN-CMM algorithm. Numerical values are rounded to 1 decimal place in this table.

values while C3 and C6 genes were similar in five out of the six Pareto solutions. It is very hard to generalize on why the evolutionary runs have converged on the limb length and constraint orientation gene values. Although it may be likely that longer limb lengths provided an evolutionary advantage in that artificial creatures with longer limbs should be able to move further distances per cycle of limb, not all of the solutions had maximal limb lengths. In fact, a number of limbs had the minimal value of 0.2cm such as *L1*, *L2* and *L8*. This combination of very small limb lengths actually resulted in entire legs that did not contribute to the locomotion of the creature upon visual inspection. This is somewhat analogous to vestigial limbs found in some animals. The presence of the non-contributing limbs may have lead to easier control requirements for the creature's legged locomotion. Screen dumps of the creature for the six global Pareto solutions are given below to provide a visualization of the evolved morphologies and the locomotion behavior generated.

Visual inspection of the locomotion behavior generated by the creatures

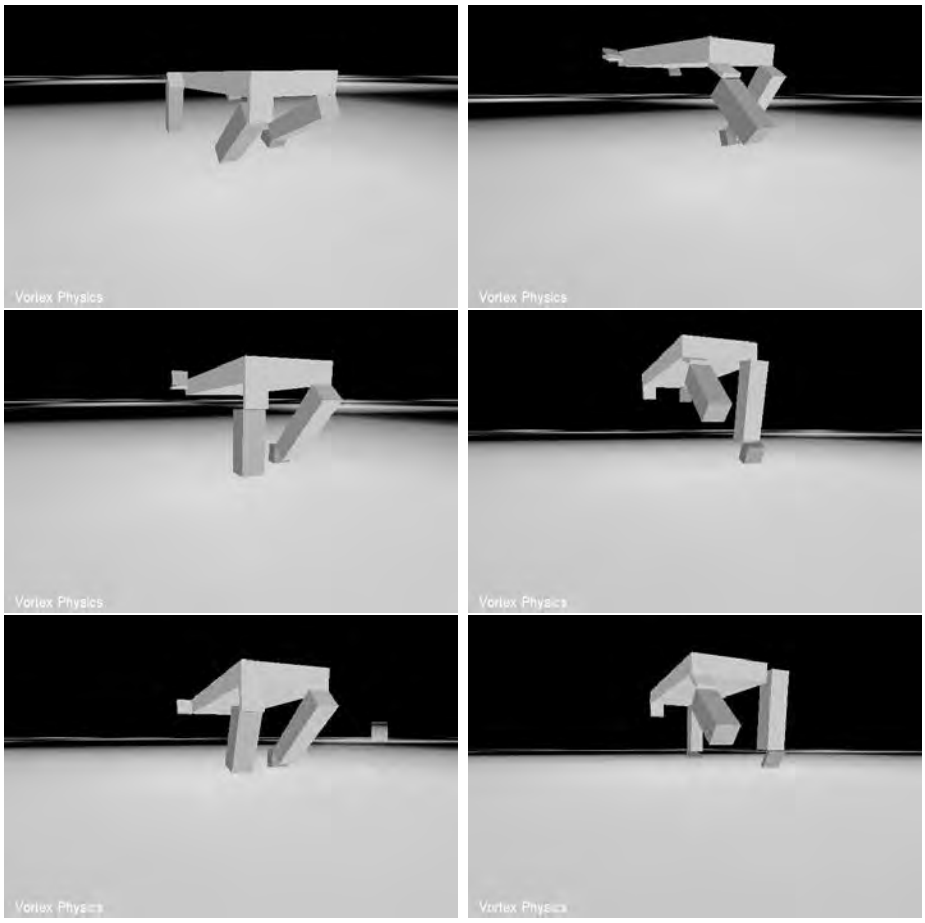


Figure 8.8: Screen dumps of the artificial creatures found on the global Pareto-frontier from co-evolving morphology and controller. 1. Solution 1 (top left), 2. Solution 2 (top right), 3. Solution 3 (middle left), 4. Solution 4 (middle right), 5. Solution 5 (bottom left), 6. Solution 6 (bottom right).

found on the global Pareto-frontier of the evolutionary runs revealed that all the creatures moved forwards by using a dynamic jumping gait rather than a statically stable walking gait (interested readers can view video clips of these evolved behaviors in the accompanying CD-ROM). Creature 1 (Figure 8.8.1) was basically a tripedal creature which generated its locomotion force from its front left, back right and back left legs while having a non-contributing front right leg. Creatures 2 through 5 (Figures 8.8.2–8.8.5) were essentially bipedal creatures that had almost identical morphologies and resultant gaits, where the locomotion force was generated by the two back legs while the two front limbs did not contribute to the forwards move-

ment. Creature 6 (Figure 8.8.6) again returned to the tripedal-like morphology seen in Creature 1. However the non-contributing leg was now switched to the front left and the locomotion force was generated by the front right, back right and back left legs. Across all the creatures, it is interesting to note that the legs which contributed to the forwards locomotion had fairly similar overall leg lengths although the individual limbs that made up the overall leg were quite different between the upper and lower limbs. Furthermore, the non-contributing limbs were fully minimized to the shortest possible length. As postulated earlier, this may somehow simplify the control requirements of the creature by reducing the number of legs that actually touched the ground during the creature's movement and hence did not require any synchronization within these legs nor coordination with other contributing legs to occur.

8.3.3 Search Space Characteristics

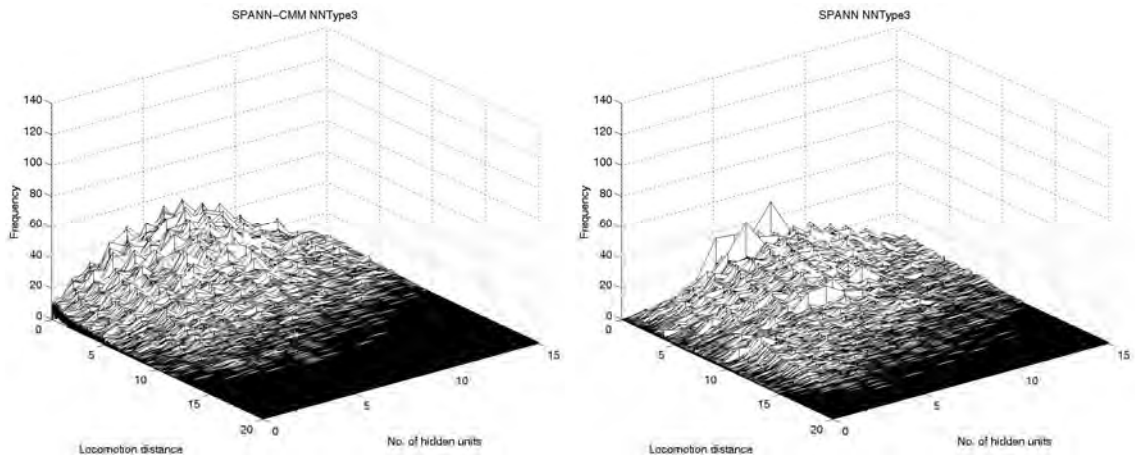


Figure 8.9: Frequency distribution of solutions obtained using the 1. SPANN-CMM (left), 2. SPANN (right) algorithms. X-axis: Locomotion distance, Y-axis: No. of hidden units, Z-axis: Frequency.

The frequency distribution of genotypes generated by SPANN-CMM across the two objective spaces were fairly uniformly spread out as depicted in Figure 8.9.1, similar to the frequency distribution obtained for SPANN (Figure 8.9.2). The

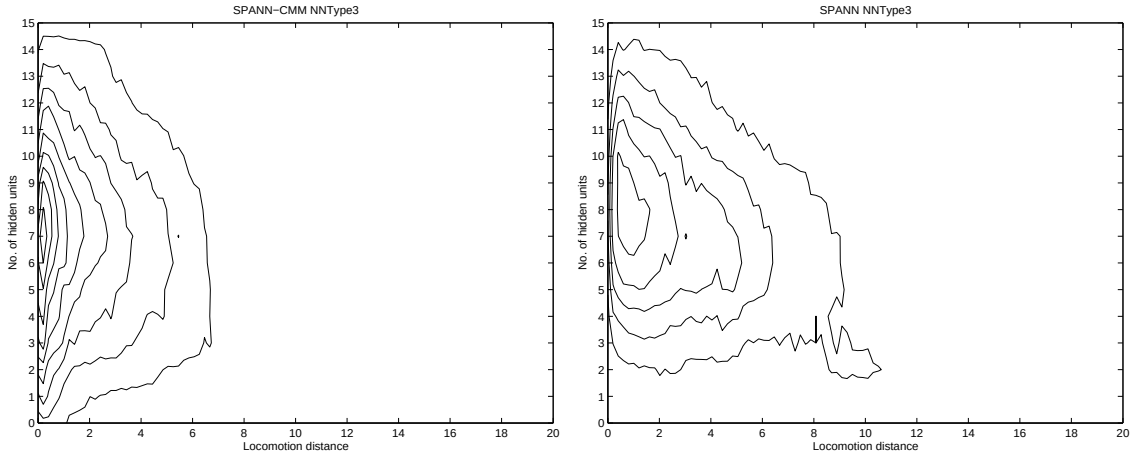


Figure 8.10: Contour graphs of frequency distribution of solutions obtained using the 1. SPANN-CMM (left), 2. SPANN (right) algorithms. X-axis: Locomotion distance, Y-axis: No. of hidden units.

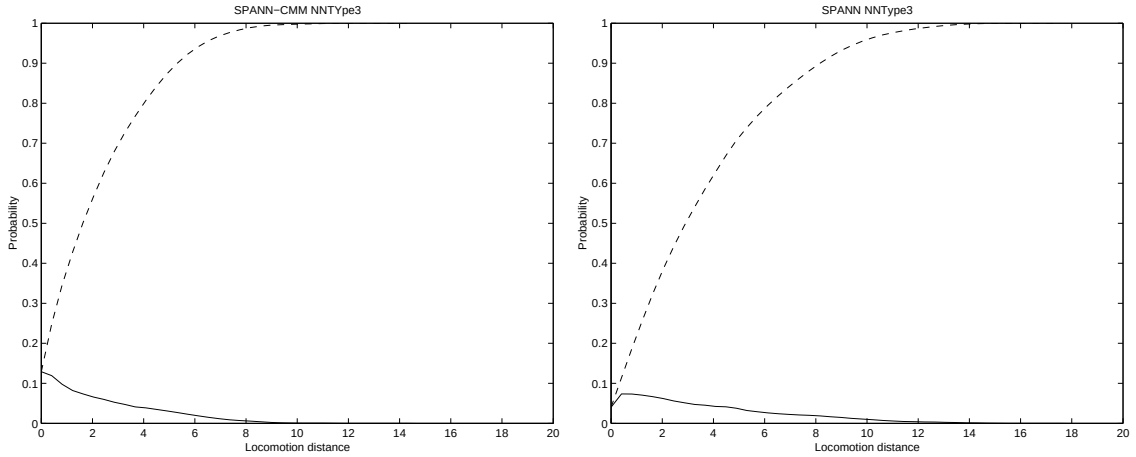


Figure 8.11: Density (solid) and cumulative (dashed) probability distribution of solutions obtained using the 1. SPANN-CMM (left), 2. SPANN (right) algorithms. X-axis: Locomotion distance, Y-axis: Probability.

contour graph in Figure 8.10.1 shows that the highest concentration of genotypes generated by SPANN-CMM used between 6 and 8 hidden units in the controller and produced very bad locomotion capabilities. Again, this can be attributed to the changing morphology of the creature, of which some may be very hard to generate good locomotion behaviors by virtue of their physical characteristics. From the

contour graphs, it can be seen that the distribution of solutions in terms of the two objectives were more spread out and less clustered within a specific region in SPANN (Figure 8.10.2) compared to SPANN-CMM, and as such was able to sample areas of the search space with higher locomotion fitness. The probability density function for SPANN-CMM shows that the probability of encountering solutions dropped to zero at around 12 units of distance (Figure 8.11.1) compared to SPANN which extended to around 14 units (Figure 8.11.2). However, the fact that SPANN-CMM was still able to produce controllers with locomotion distances higher than SPANN in spite of having a higher concentration of solutions in the lower fitness regions of the objective space shows that the inclusion of morphological parameters for evolution is not entirely counterproductive but can in fact find good combinations of controller and morphology.

8.4 Chapter Summary

We have investigated the co-evolution of morphology and mind by augmenting the SPANN algorithm to allow for simultaneous optimization of both the creature's body and controller. Certain morphological parameters which were previously constrained have been relaxed to allow for the ANN controller to be optimized while at the same time allowing evolution to find suitable morphologies that would work well with the controllers. It was found that although no significant improvement in locomotion distance was achieved, significantly different locomotion behaviors emerged together with radically different body designs. Dynamic locomotion gaits based on a jumping motion generated mainly from hind legs were found in creatures that were essentially bipedal and tripedal in their legged locomotion. A characterization of the different solutions showed that the creatures existing on the global Pareto-frontier had similar complexities in terms of both control and locomotion behavior.

Chapter 9

Conclusion

"And in the future? Who knows? But it seems almost certain that the first forms of alien life we see will not be through telescopes, but through the windows of our computer screens into digital universes. The first person to hold a conversation with an alien intelligence will not be an astronaut: it will be a computer scientist or computational neuroscientist, talking to an evolved digital neural network. The first glimpses of non-human cultures and technologies will occur in our research labs, where the digital biology grows more complex day by day." (p.14)

(**Peter J. Bentley, 2002**)

9.1 Summary of Results

In this thesis, we presented a systematic study of evolving ANN controllers for the legged locomotion of virtual organisms using a Pareto multi-objective evolutionary optimization approach. A virtual and physically accurate world was created to simulate the evolution of locomotion behavior in a quadruped creature. A self-adaptive Pareto EMO algorithm called SPANN, which allowed for the generation of Pareto solutions that optimized multiple objectives distinctly, was implemented to evolve the ANN-based control mechanism for the quadruped. The search spaces un-

derlying four classes of ANNs with different connectivity types were characterized. SPANN was then used to evolve Pareto solutions that maximized the locomotion distance of a fixed morphology quadruped and minimized usage of hidden units in its ANN controller. More conventional methods of evolutionary optimization were subsequently compared against SPANN to ascertain the true advantages offered by the self-adaptive Pareto EMO methodology. An approach for the characterization of the morphological and behavioral complexity was then proposed based on a multi-objective viewpoint. Finally, the simultaneous evolution of morphology and controller was conducted by relaxing some of the morphological parameters imposed on the artificial creature to explore the different creature designs that can be found through the co-evolutionary optimization methods.

The main findings from the investigations carried out in this thesis are as follows:

1. Search space characterization of four different types of ANN controller architectures (NNType0, NNType1, NNType2 and NNType3) using random search, hill-climbing and random walk showed no significant differences in terms of the ease of finding good quality locomotion controllers.

2. The fitness landscape of the evolutionary search spaces had both rugged and smooth sections depending on the sub-spaces being explored. The variety of rugged shapes on the landscape was high indicating that epistatic interactions between genes in the genotype were high. A correspondingly high degree of modality in the fitness landscape was also noted.

3. The solution space was found to be highly heterogeneous. A uniform sampling of the genotype space yielded a highly skewed distribution of solutions in the objective space.

4. An EMO algorithm called SPANN was implemented for the multi-objective evolution of artificial creature controllers. Artificial creature controllers were successfully evolved for minimum hidden layer size and maximum horizontal locomotion distance using four different types of ANN architecture. Although

the overall best solution was found using the NNType3 architecture, statistical tests showed no significant difference existed between the results obtained.

5. A comparison between SPANN and random search, hill-climbing and random walk showed that the evolutionary search implemented in SPANN produced statistically superior solutions.

6. Recurrent connections did not provide any significant advantages over conventional feed-forward neural network architectures for evolving locomotion behavior in a quadruped.

7. Pure reactive agents not requiring hidden layer transformations in the ANN controller produced sufficiently good locomotion capabilities. The use of direct input-output connections in a perceptron-like controller was sufficient for generating a basic locomotion behavior in the quadruped.

8. The SPANN algorithm discovered reasonably good quality controllers but required significantly less overall computational costs compared to a single-objective EA, a weighted sum EMO algorithm as well as a hand-tuned EMO algorithm. The controllers evolved using SPANN were comparable to those obtained with NSGA-II, one of the current state-of-the-art Pareto EMO algorithms. The self-adaptive Pareto EMO approach implemented in the SPANN algorithm provided significant advantages over conventional evolutionary optimization algorithms by: (1) reducing the number of runs required to test different design factors associated with the synthesis of artificial creatures, (2) preserving genetic diversity, and (3) offering extra-dimensional bypasses for the search process to reach fitter solution spaces.

9. The overall best locomotion controller evolved using SPANN had the least amount of redundancy present in the ANN compared to the overall best locomotion controllers evolved using a hand-tuned EMO algorithm, a weighted sum EMO algorithm and a single-objective EA.

10. Some level of coordination and synchronization was achieved by the overall

best locomotion controller evolved using SPANN. The controller continued to perform well despite the presence of low levels of noise in the quadruped's sensors and actuators.

11. Taking a multi-objective view towards complexity provided a useful platform for comparing between the evolved behavioral and morphological complexities of embodied creatures. A Pareto-based methodology for characterizing complexity was implemented and using this approach, it was found that the morphological complexity of a hexapod was higher than a quadruped while the behavioral complexity of a quadruped was higher than a hexapod.

12. Significantly different locomotion behaviors emerged together with radically different body designs when certain morphological constraints were relaxed to allow for co-evolutionary optimization of morphology and controller to occur in the SPANN-CMM algorithm. Dynamic locomotion gaits based on a jumping motion generated mainly from hind legs were found in creatures that were essentially bipedal and tripedal in locomotion behavior. A comparison between solutions evolved with and without co-evolution of morphology showed that creatures existing on the global Pareto-frontier of both evolutionary systems had similar complexities in terms of both controller requirements and locomotion behavior.

9.2 Future Work

Numerous avenues for further explorations and investigations have emerged from this body of work. Some open research questions have already been highlighted in the respective chapters where they directly followed on from the work completed in the experiments. Here we outline more diverse and philosophical future research directions that encompass the evolution of artificial organisms at a higher level.

Other forms of control mechanism can be used in place of these basic ANNs. On a lower level, CPGs and other types of algorithmic controllers would certainly be also useful to evolve for simpler artificial organisms. On a higher level, more ad-

vanced forms of ANNs such as those having higher-order activation functions, GasNets that use temporally and spatially adaptive neurons (Husbands, Smith, Jakobi, and O'Shea 1998), as well as Pulsed Neural Networks that use spiking neurons (Floreano and Mattiussi 2001) may allow for the emergence of more complex behaviors in artificial organisms that are required to perform more complicated tasks and in changing environments.

The power of a Pareto multi-objective approach lies in the flexibility and ease of incorporating new objectives and elements into the artificial evolutionary process. The inclusion of elements such as compactness of genetic material in EAs that utilize variable length chromosomes or other more elaborate developmental encodings as a distinct and separate objective on top of the primary objective will not only provide useful ways of improving the efficiency of the EA but may possibly also provide interesting insights into why vastly different genome lengths are found in biological organisms. Other elements that will be fruitful to investigate as separate evolutionary objectives from an artificial life perspective include phylogenetic diversity, number of body parts/joints and physical energy consumption to name but a few.

The SPANN algorithm can be beneficial in evolving controllers not only for legged robots but also wheeled and other forms of physical robots. Again, the multi-objectivity of the artificial evolution can easily incorporate additional engineering factors such as noise and stress tolerance into the optimization process. It will also be interesting to expand the SPANN-CMM algorithm to allow for fully and freely evolvable robotic forms that are not based on any underlying body plan and evolved from very basic structures, perhaps even at the atomic level. This can have far-reaching implications on the evolution of minimal controllers and morphologies of recyclable micro-machines that can be created with nanotechnology and evolvable hardware. The fully automated design, fabrication and re-use cycle of such evolvable systems would then truly constitute a form of artificial life. On an even grander scale, although representing a scenario which perhaps demands more serious ethical discussions, such artificial creatures can at later generations even fabricate micro-

factories capable of re-designing and evolving existing creatures into more efficient and effective forms of life and intelligence.

9.3 Concluding Remarks

The automatic synthesis of embodied and situated artificial creatures that are fully autonomous and intelligent is a truly lofty goal. Significant advancements have been made through numerous artificial evolutionary studies and in this thesis, we have proposed the use of a Pareto multi-objective approach to this end. We have shown that multiple objectives can be treated as distinct and separate optimization goals in a Pareto EMO algorithm when evolving such organisms. This body of work should not be seen as merely an attempt to present a new artificial life system or an evolutionary robotics algorithm, although it does exhibit many useful characteristics in these respects, but as a new paradigm in which evolutionary computation can be used in a truly purposeful and powerful way in terms of designing, generating and synthesizing intelligent machines and artifacts that do not only mindlessly avoid walls or react to our facial expressions but are able to exhibit a host of other intelligent behaviors that constitute the multi-objective nature of our world. This technology will pave the way for such intelligent creatures to be realized in the not-too-distant future, which will then truly revolutionize the way in which we humans think about and live life.

Bibliography

Abbass, H. A. (2001). A memetic Pareto evolutionary approach to artificial neural networks. In M. Stumptner, D. Corbett, and M. Brooks (Eds.), *Proceedings of the 14th International Joint Conference on Artificial Intelligence (AI 2001)*, Volume 2256 of *Lecture Notes in Artificial Intelligence (LNAI)*, pp. 1–12. Berlin: Springer-Verlag.

Abbass, H. A. (2002a). An evolutionary artificial neural network approach for breast cancer diagnosis. *Artificial Intelligence in Medicine 25*(3), 265–281.

Abbass, H. A. (2002b). The self-adaptive Pareto differential evolution algorithm. In *Proceedings of the 2002 Congress on Evolutionary Computation (CEC2002)*, Volume 1, pp. 831–836. Piscataway, NJ: IEEE Press.

Abbass, H. A. (2003). Speeding up back-propagation using multiobjective evolutionary algorithms. *Neural Computation (Accepted to appear)*.

Abbass, H. A. and K. Deb (2003). Searching under multi-evolutionary pressures. In C. Fonseca, P. Fleming, E. Zitzler, K. Deb, and L. Thiele (Eds.), *Proceedings of the Second International Conference on Evolutionary Multi-Criterion Optimization (EMO 2003)*, Volume 2632 of *Lecture Notes in Computer Science (LNCS)*, pp. 391–404. Berlin: Springer-Verlag.

Abbass, H. A. and R. Sarker (2002). The Pareto differential evolution algorithm. *International Journal on Artificial Intelligence Tools 11*(4), 531–552.

Abbass, H. A., R. Sarker, and C. Newton (2001). PDE: A Pareto-frontier differential evolution approach for multi-objective optimization problems. In *Proceedings of the 2001 Congress on Evolutionary Computation (CEC2001)*, Vol-

ume 2, pp. 971–978. Piscataway, NJ: IEEE Press.

Adami, C. (1998). *Introduction to Artificial Life*. Santa Clara, CA: Telos Publishing.

Albin, P. S. (1980). The complexity of social groups and social systems described by graph structures. *Mathematical Social Sciences 1*(1), 101–129.

Andrews, G. and G. S. Halford (2002). A cognitive complexity metric applied to cognitive development. *Cognitive Psychology 45*(2), 153–219.

Angeline, P. J. and J. B. Pollack (1993). Competitive environments evolve better solutions for complex tasks. In S. Forrest (Ed.), *Proceedings of the Fifth International Conference on Genetic Algorithms*, pp. 264–270. San Mateo, CA: Morgan Kaufmann.

Angeline, P. J., G. M. Saunders, and J. B. Pollack (1994). An evolutionary algorithm that constructs recurrent neural networks. *IEEE Transactions on Neural Networks 5*(1), 54–65.

Arkin, R. C. (1998). *Behavior-Based Robotics*. Cambridge, MA: MIT Press.

Arnold, D. (1997). Evolution of legged locomotion. Unpublished masters thesis, School of Computing Science, Simon Fraser University, Burnaby, Canada.

Atlan, H. and M. Koppel (1990). The cellular computer DNA: Program or data. *Bulletin of Mathematical Biology 52*(3), 335–348.

Aurelius, M. (167). *The Meditations*, Chapter 6, verse 5. Excerpt from (Bartlett 1992).

Auyang, S. Y. (1998). *Foundations of Complex-System Theories: In Economics, Evolutionary Biology and Statistical Physics*. Cambridge, UK: Cambridge University Press.

Badii, R. and A. Politi (1997). *Complexity: Hierarchical Structures and Scaling in Physics*. Cambridge, UK: Cambridge University Press.

Barnett, L. (1998). Ruggedness and neutrality: The NKp family of fitness landscapes. In C. Adami, R. Belew, H. Kitano, and C. Taylor (Eds.), *Artificial*

Life VI: Proceedings of the Sixth International Conference on the Simulation and Synthesis of Living Systems, pp. 18–27. Cambridge, MA: MIT Press.

Bartlett, J. (1992). In J. Kaplan (Ed.), *Bartlett's Familiar Quotations*. New York: Little, Brown & Company.

Beer, R. D. and J. C. Gallagher (1992). Evolving dynamical neural networks for adaptive behavior. *Adaptive Behavior 1*(1), 91–122.

Belew, R. K., J. McInerney, and N. N. Schraudolph (1992). Evolving networks: Using the genetic algorithm with connectionist learning. In C. Langton, C. Taylor, J. Farmer, and S. Rasmussen (Eds.), *Artificial Life II: Proceedings of the Second International Workshop on the Synthesis and Simulation of Living Systems*, pp. 511–547. Redwood City, CA: Addison-Wesley.

Bennett, C. H. (1988). Logical depth and physical complexity. In R. Herken (Ed.), *Universal Turing Machine, A Half-Century Survey*, pp. 227–257. Oxford: Oxford University Press.

Bentley, P. J. (2002). *Digital Biology: How Nature is Transforming Our Technology and Our Lives*. New York: Simon and Schuster.

Bongard, J. C. (2002a). Evolved sensor fusion and dissociation in an embodied agent [online]. In *Proceedings of the EPSRC/BBSRC International Workshop on Biologically-Inspired Robotics: The Legacy of W. Grey Walter*, Bristol, UK, pp. 102–109. http://www.ifi.unizh.ch/ailab/people/bongard/ [cited - 10 August 2002].

Bongard, J. C. (2002b). Evolving modular genetic regulatory networks. In *Proceedings of the 2002 Congress on Evolutionary Computation (CEC2002)*, pp. 1872–1877. Piscataway, NJ: IEEE Press.

Bongard, J. C. and C. Paul (2000). Investigating morphological symmetry and locomotive efficiency using virtual embodied evolution. In J.-A. Meyer, A. Berthoz, D. Floreano, H. Roitblat, and S. Wilson (Eds.), *From Animals to Animats 6: Proceedings of the Sixth International Conference on the Simulation of Adaptive Behavior (SAB2000)*, pp. 420–429. Cambridge, MA: MIT

Press.

Bongard, J. C. and C. Paul (2001). Making evolution an offer it can't refuse: Morphology and the extradimensional bypass. In J. Keleman and P. Sosik (Eds.), *Advances in Artificial Life: Proceedings of the 6th European Conference on Artificial Life (ECAL01)*, pp. 401–412. Berlin: Springer-Verlag.

Bongard, J. C. and R. Pfeifer (2001). Repeated structure and dissociation of genotypic and phenotypic complexity in artificial ontogeny. In L. Spector, E. Goodman, A. Wu, W. Langdon, H. Voight, M. Gen, S. Sen, M. Dorigo, S. Pezeshk, M. Garzon, and E. Burke (Eds.), *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2001)*, pp. 829–836. San Francisco: Morgan Kaufmann.

Bongard, J. C. and R. Pfeifer (2002). A method for isolating morphological effects on evolved behavior. In B. Hallam, D. Floreano, G. Hayes, J. Hallam, and J.-A. Meyer (Eds.), *From Animals to Animats 7: Proceedings of the Seventh International Conference on the Simulation of Adaptive Behavior (SAB2002)*, pp. 305–311. Cambridge, MA: MIT Press.

Bonner, J. T. (1988). *The Evolution of Complexity by Means of Natural Selection*. Princeton, NJ: Princeton University Press.

Brooks, R. A. (1995). Intelligence without reason. In L. Steels and R. Brooks (Eds.), *The Artificial Life Route to Artificial Intelligence: Building Embodied, Situated Agents*, Chapter 2, pp. 25–81. Hillsdale, NJ: Lawrence Erlbaum Associates Publishers.

Butts, C. T. (2001). The complexity of social networks: Theoretical and empirical findings. *Social Networks 23*(1), 31–71.

Cariani, P. (1992). Emergence and artificial life. In C. Langton, C. Taylor, J. Farmer, and S. Rasmussen (Eds.), *Artificial Life II: Proceedings of the Second International Workshop on the Synthesis and Simulation of Living Systems*, pp. 775–797. Redwood City, CA: Addison-Wesley.

Casti, J. L. (1986). On system complexity: Identification, measurement and man-

agement. In J. Casti and A. Karlqvist (Eds.), *Complexity, Language and Life: Mathematical Approaches*, Chapter 6, pp. 146–173. Berlin: Springer-Verlag.

Cavalier-Smith, T. (1985). *The Evolution of Genome Size*. New York: Wiley.

Clement, M. B. (1999). Analyst forecast accuracy: Do ability, resources, and portfolio complexity matter? *Journal of Accounting and Economics 27*(3), 285–303.

Cliff, D., I. Harvey, and P. Husbands (1993). Explorations in evolutionary robotics. *Adaptive Behavior 2*(1), 73–110.

Cliff, D. and G. F. Miller (1996). Co-evolution of pursuit and evasion II: Simulation methods and results. In P. Maes, M. Mataric, J.-A. Meyer, J. Pollack, and S. Wilson (Eds.), *From Animals to Animats 4: Proceedings of the Fourth International Conference on the Simulation of Adaptive Behavior (SAB96)*, pp. 506–515. Cambridge, MA: MIT Press.

CM Labs (2002). Vortex [online]. http://www.cm-labs.com [cited - 25 January 2002].

CM Labs Press Release (2003). Critical mass labs bound for mars [online]. http://www.cm-labs.com/news/releases/cml3.php [cited - 7 March 2003].

Coello Coello, C. A., A. D. Christiansen, and A. H. Aguirre (1998). Using a new GA-based multiobjective optimization technique for the design of robot arms. *Robotica 16*, 401–414.

Coello Coello, C. A., D. A. Van Veldhuizen, and G. B. Lamont (2002). *Evolutionary Algorithms for Solving Multi-Objective Problems*. New York: Kluwer Academic.

Cooper, C. (1993). Complexity in C3I systems. In D. Green and T. Bossomaier (Eds.), *Complex Systems: From Biology to Computation*, pp. 223–231. Amsterdam: IOS Press.

Coveney, P. and R. Highfield (1995). *Frontiers of Complexity: The Search for Order in a Chaotic World*. New York: Fawcett Columbine.

Dautenhahn, K. (1996). Embodied cognition in animals and artifacts. In M. Mataric (Ed.), *Embodied Cognition and Action: Papers from the 1996 AAAI Fall Synposium*, AAAI Technical Report FS–96–02, pp. 27–32. Menlo Park, CA: AAAI Press.

Dautenhahn, K. (1999). Embodiment and interaction in socially intelligent life-like agents. In C. Nehaniv (Ed.), *Computation for Metaphors, Analogy and Agent*, Volume 1562 of *Lecture Notes in Artificial Intelligence (LNAI)*, pp. 102–142. Berlin: Springer-Verlag.

Deb, K. (2001). *Multi-objective Optimization using Evolutionary Algorithms*. Chicester, UK: John Wiley & Sons.

Deb, K., S. Agrawal, A. Pratab, and T. Meyarivan (2000). A fast elitist non-dominated sorting genetic algorithm for multi-objective optimization: NSGA-II. In M. Schoenauer, K. Deb, G. Rudolph, X. Yao, E. Lutton, J. Merelo, and H.-P. Schwefel (Eds.), *Proceedings of the Parallel Problem Solving from Nature VI Conference (PPSN VI)*, pp. 849–858. Berlin: Springer-Verlag.

Deb, K. and J. Horn (2000). Introduction to the special issue: Multicriterion optimization. *Evolutionary Computation 8*(2), iii–iv.

Deb, K., A. Pratab, S. Agrawal, and T. Meyarivan (2002). A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation 6*(2), 182–197.

Dellaert, F. and R. D. Beer (1994). 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.

DeShazo, J. and G. Fermo (2002). Designing choice sets for stated preference methods: The effects of complexity on choice consistency. *Journal of Environmental Economics and Management 44*(1), 123–143.

Dittrich, P., A. Skusa, W. Banzhaf, and W. Kantschik (1999). Dynamical properties of the fitness landscape of a GP controlled random morphology robot.

In W. Banzhaf, J. Daida, A. Eiben, M. Garzon, V. Hanovar, M. Jakiela, and R. Smith (Eds.), *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-99)*, Volume 2, pp. 1002–1008. San Francisco: Morgan Kaufmann.

Dozier, G., S. McCullough, A. Homaifar, E. Tunstel, and L. Moore (1998). Multi-objective evolutionary path planning via fuzzy tournament selection. In *Proceedings of the IEEE International Conference on Evolutionary Computation (ICEC'98)*, pp. 684–689. Piscataway, NJ: IEEE Press.

Edmonds, B. (1997). From complexity to agent modelling and back again - some implications for economics [online]. In *Workshop on Economics and the Sciences of Complexity*, Brussels, Belgium. http://cfpm.org/cpmrep24.html [cited - 17 October 2002].

Edmonds, B. (1999). *Syntactic Measures of Complexity*. Unpublished PhD thesis, University of Manchester, England.

Eggenberger, P. (1996). Cell interactions as a control tool of developmental processes for evolutionary robotics. In P. Maes, M. Mataric, J.-A. Meyer, J. Pollack, and S. Wilson (Eds.), *From Animals to Animats 4: Proceedings of the Fourth International Conference on the Simulation of Adaptive Behavior (SAB96)*, pp. 440–448. Cambridge, MA: MIT Press.

Eggenberger, P. (1997). Evolving morphologies of simulated 3D organisms based on differential gene expression. In P. Husbands and I. Harvey (Eds.), *Advances in Artificial Life: Proceedings of the 4th European Conference on Artificial Life (ECAL97)*, pp. 205–213. Cambridge, MA: MIT Press.

Eggenberger, P., G. Gomez, and R. Pfeifer (2002). Evolving the morphology of a neural network for controlling a foveating retina. In R. Standish, M. Bedau, and H. Abbass (Eds.), *Artificial Life VIII: The 8th International Conference on Artificial Life*, pp. 243–251. Cambridge, MA: MIT Press.

Elman, J. L. (1990). Finding structure in time. *Cognitive Science 14* (2), 179–211.

Emmeche, C. (1994). *Garden in the Machine*. Princeton, NJ: Princeton University Press.

EnsEMBL.Org (2002). Ensembl genome browser [online]. http://www.ensembl.org [cited - 6 November 2002].

Feldman, D. P. and J. P. Crutchfield (1998a). Discovering noncritical organization: Statistical mechanical, information theoretic, and computational views of patterns in one-dimensional spin systems. Working Paper 98-04-026, Sante Fe Institute.

Feldman, D. P. and J. P. Crutchfield (1998b). Measures of statistical complexity: Why? *Physics Letters A 238*, 244–252.

Floreano, D. (1998). Evolutionary robotics in artificial life and behavior engineering. In T. Gomi (Ed.), *Proceedings of the 6th International Symposium on Evolutionary Robotics: From Intelligent Robots to Artificial Life (ER'98)*, pp. 77–104. Ontario: AAI Books.

Floreano, D. and C. Mattiussi (2001). Evolution of spiking neural controllers for autonomous vision-based robots. In T. Gomi (Ed.), *Proceedings of the 8th International Symposium on Evolutionary Robotics: From Intelligent Robots to Artificial Life (ER2001)*, pp. 38–61. Berlin: Springer-Verlag.

Floreano, D. and F. Mondada (1998). Evolutionary neurocontrollers for autonomous mobile robots. *Neural Networks 11*, 1461–1478.

Floreano, D., S. Nolfi, and F. Mondada (2001). Co-evolution and ontogenetic change in competing robots. In M. Patel, V. Honavar, and K. Balakrishnan (Eds.), *Advances in the Evolutionary Synthesis of Intelligent Agents*, pp. 273–306. Cambridge, MA: MIT Press.

Floreano, D., N. Schoeni, G. Caprari, and J. Blynel (2002). Evolutionary bits 'N' spikes. In R. Standish, M. Bedau, and H. Abbass (Eds.), *Artificial Life VIII: The 8th International Conference on Artificial Life*, pp. 333–344. Cambridge, MA: MIT Press.

Floreano, D. and J. Urzelai (1998). Evolution and learning in autonomous mobile robots. In D. Mange and M. Tomassini (Eds.), *Bio-Inspired Computing Machines: Towards Novel Computational Architectures* (1st ed.)., Chapter 12, pp. 317–364. Lausanne, Switzerland: Presses Polytechniques Et Universitaires Romandes.

Floreano, D. and J. Urzelai (2000). Evolutionary robotics: The next generation. In T. Gomi (Ed.), *Proceedings of the 7th International Symposium on Evolutionary Robotics: From Intelligent Robots to Artificial Life (ER2000)*, pp. 231–266. Ontario: AAI Books.

Floreano, D. and J. Urzelai (2001). Neural morphogenesis, synaptic plasticity, and evolution. *Theory in Biosciences 120*(3–4), 225–240.

Fogel, D. B., L. J. Fogel, and V. W. Porto (1990). Evolving neural networks. *Biological Cybernetics 63*(6), 487–493.

Fonseca, C. M. and P. J. Fleming (1993). Genetic algorithms for multiobjective optimization: Formulation, discussion and generalization. In S. Forrest (Ed.), *Proceedings of the Fifth International Conference on Genetic Algorithms*, pp. 416–423. San Mateo, CA: Morgan Kaufmann.

Foster, D., J. McCullagh, and T. Whitford (1999). Evolution versus training: An investigation into combining genetic algorithms and neural networks. In T. Gedeon, P. Wong, S. Halgamuge, N. Kasabov, D. Nauck, and K. Fukushima (Eds.), *Proceedings of the 6th International Conference on Neural Information Processing (ICONIP'99)*, Volume 3, pp. 848–854. Piscataway, NJ: IEEE Press.

Franklin, S. P. (1995). *Artificial Minds*. Cambridge, MA: MIT Press.

Fujii, A., A. Ishiguro, T. Aoki, and P. Eggenberger (2001). Evolving bipedal locomotion with a dynamically-rearranging neural network. In J. Keleman and P. Sosik (Eds.), *Advances in Artificial Life: Proceedings of the 6th European Conference on Artificial Life (ECAL01)*, pp. 507–518. Berlin: Springer-Verlag.

Gacogne, L. (1997). Research of Pareto set by genetic algorithm, application to multicriteria optimization of fuzzy controller [online]. In *Proceed-*

ings of the 5th European Congress on Intelligent Techniques and Soft Computing (EUFIT'97), Aachen, Germany, pp. 837–845. http://www.jeo.org/emo/EMOOconferences.html [cited - 8 January 2003].

Gacogne, L. (1999). Multiple objective optimization of fuzzy rules for obstacles avoiding by an evolution algorithm with adaptative operators [online]. In *Proceedings of the Fifth International Mendel Conference on Soft Computing (Mendel'99)*, Brno, Czech Republic, pp. 236–242. http://www.jeo.org/emo/EMOOconferences.html [cited - 8 January 2003].

Gallagher, J. C., R. D. Beer, K. S. Espenschied, and R. D. Quinn (1996). Application of evolved locomotion controllers to a hexapod robot. *Robotics and Autonomous Systems 19*, 95–103.

Garey, M. R. and D. S. Johnson (1979). *Computers and Intractability : A Guide to the Theory of NP-Completeness*. San Francisco: W.H. Freeman.

Gibas, C. and P. Jambeck (2001). *Developing Bioinformatics Skills*. Sebastopol, CA: O'Reilly.

Gibson, E. (1998). Linguistic complexity: Locality of syntactic dependencies. *Cognition 68*(1), 1–76.

Goldberg, D. E. (1989). *Genetic Algorithms in Search, Optimization and Machine Learning*. Reading, MA: Addison-Wesley.

Gomi, T. and K. Ide (1998). Evolution of gaits of a legged robot. In *Proceedings of the 1998 IEEE World Congress on Computational Intelligence (WCCI'98)*, Volume 1, pp. 159–164. Piscataway, NJ: IEEE Press.

Grand, S. (2001). *Creation: Life and How to Make It*. London: Phoenix.

Gritz, L. and J. K. Hahn (1997). Genetic programming evolution of controllers for 3-D character animation. In J. Koza, K. Deb, M. Dorigo, D. Fogel, M. Garzon, H. Iba, and R. Riolo (Eds.), *Genetic Programming 1997: Proceedings of the 2nd Annual Conference*, pp. 139–146. San Francisco: Morgan Kaufmann.

Gruau, F. (1994). *Neural Network Synthesis using Cellular Encoding and the Ge-*

netic Algorithm. Unpublished PhD thesis, Laboratoire de l'Informatique du Parallilisme, Ecole Normale Supirieure de Lyon, France.

Gruau, F. (1997). Cellular encoding for interactive evolutionary robotics. In P. Husbands and I. Harvey (Eds.), *Advances in Artificial Life: Proceedings of the 4th European Conference on Artificial Life (ECAL97)*, pp. 368–377. Cambridge, MA: MIT Press.

Grzimek, B. (1984). Grzimek's animal life encyclopedia. In *Insects*, Volume 2. New York: Van Nostrand Reinhold.

Halford, G. S., W. H. Wilson, and S. Phillips (1998). Processing capacity defined by relational complexity: Implications for comparative, developmental, and cognitive psychology. *Behaviorial and Brain Sciences 21*(6), 803–831.

Harvey, I. (1992). Species adaptation genetic algorithm: The basis for a continuing SAGA. In F. Varela and P. Bourgine (Eds.), *Proceedings of the First European Conference on Artificial Life: Towards a Practice of Autonomous Systems*, pp. 346–354. Cambridge, MA: MIT Press.

Harvey, I. (1997). Artificial evolution and real robots. *Artificial Life and Robotics 1*(1), 35–38.

Harvey, I., P. Husbands, and D. Cliff (1994). Seeing the light: Artificial evolution, real vision. In D. Cliff, P. Husbands, J.-A. Meyer, and S. Wilson (Eds.), *From Animals to Animats 3: Proceedings of the Third International Conference on the Simulation of Adaptive Behavior (SAB94)*, pp. 392–401. Cambridge, MA: MIT Press.

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

Haykin, S. (1999). *Neural Networks: A Comprehensive Foundation* (2nd ed.). Upper Saddle River, NJ: Prentice-Hall.

Hillis, D. W. (1992). Co-evolving parasites improve simulated evolution as an optimization procedure. In C. Langton, C. Taylor, J. Farmer, and S. Rasmussen

(Eds.), *Artificial Life II: Proceedings of the Second International Workshop on the Synthesis and Simulation of Living Systems*, pp. 313–324. Redwood City, CA: Addison-Wesley.

Holm, H. J. (1993). *Complexity in Economic Theory: Automata Theoretical Approach*. Unpublished PhD thesis, Lund University, Lund, Sweden.

Hordijk, W. (1996). A measure of landscapes. *Evolutionary Computation 4*(4), 335–360.

Horn, J., N. Nafpliotis, and D. E. Goldberg (1994). A niched Pareto genetic algorithm for multiobjective optimization. In *Proceedings fo the First IEEE Conference on Evolutionary Computation*, Volume 1, pp. 82–87. Piscataway, NJ: IEEE Press.

Hornby, G. S., M. Fujita, S. Takamura, T. Yamamoto, and O. Hanagata (1999). Autonomous evolution of gaits with the Sony quadruped robot. In W. Banzhaf, J. Daida, A. Eiben, M. Garzon, V. Hanovar, M. Jakiela, and R. Smith (Eds.), *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-99)*, Volume 2, pp. 1297–1304. San Francisco: Morgan Kaufmann.

Hornby, G. S., H. Lipson, and J. B. Pollack (2001). Evolution of generative design systems for modular physical robots. In *Proceedings of the International Conference on Robotics and Automation (IRCA 2001)*, Volume 4, pp. 4146–4151. Piscataway, NJ: IEEE Press.

Hornby, G. S. and J. B. Pollack (2001a). Body-brain coevolution using L-systems as a generative encoding. In L. Spector, E. Goodman, A. Wu, W. Langdon, H. Voight, M. Gen, S. Sen, M. Dorigo, S. Pezeshk, M. Garzon, and E. Burke (Eds.), *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2001)*, pp. 868–875. San Francisco: Morgan Kaufmann.

Hornby, G. S. and J. B. Pollack (2001b). Evolving L-Systems to generate virtual creatures. *Computers and Graphics 25*(6), 1041–1048.

Hornby, G. S. and J. B. Pollack (2002). Creating high-level components with a generative representation for body-brain evolution. *Artificial Life 8*(3), 223–

246.

Hornby, G. S., S. Takamura, J. Yokono, O. Hanagata, T. Yamamoto, and M. Fujita (2000). Evolving robust gaits with AIBO. In *Proceedings of the International Conference on Robotics and Automation (IRCA 2000)*, Volume 3, pp. 3040–3045. Piscataway, NJ: IEEE Press.

Hulse, M., B. Lara, F. Pasemann, and U. Steinmetz (2001). Evolving neural behavior control for autonomous robots. In *Proceedings of the International Conference on Artificial Neural Networks (ICANN 2001)*, pp. 957–962. Berlin: Springer-Verlag.

Husbands, P., I. Harvey, N. Jakobi, A. Thompson, and D. Cliff (1997). Evolutionary robotics. In T. Back, D. Fogel, and Z. Michalewicz (Eds.), *Handbook of Evolutionary Computation*, Chapter 3.7, pp. 1–11. Bristol, Philadelphia: Institute of Physics Publishing.

Husbands, P. and F. Mill (1991). Simulated co-evolution as the mechanism for emergent planning and scheduling. In R. K. Belew and L. B. Booker (Eds.), *Proceedings of the Fourth International Conference on Genetic Algorithms*, pp. 264–270. San Mateo, CA: Morgan Kaufmann.

Husbands, P., A. Philippides, T. Smith, and M. O'Shea (2001). Volume signalling in real and robot nervous systems. *Theory in Biosciences 120*(3–4), 253–269.

Husbands, P., T. Smith, N. Jakobi, and M. O'Shea (1998). Better living through chemistry: Evolving GasNets for robot control. *Connection Science 10*(3–4), 185–210.

Huynen, M. A. (1996). Exploring phenotype space through neutral evolution. *Journal of Molecular Evolution 43*, 165–169.

Ijspeert, A. J. (1999). Synthetic approaches to neurobiology: Review and case study in the control of anguiliform locomotion. In D. Floreano, J.-D. Nicoud, and F. Mondada (Eds.), *Advances in Artificial Life: Proceedings of the 5th European Conference on Artificial Life (ECAL99)*, pp. 195–204. Berlin: Springer-Verlag.

Ijspeert, A. J. (2000). A 3-D biomechanical model of the salamander. In J.-C. Heudin (Ed.), *Proceedings of the 2nd International Conference on Virtual Worlds (VW 2000)*, pp. 225–234. Berlin: Springer-Verlag.

Ijspeert, A. J. (2001). A connectionist central pattern generator for the aquatic and terrestrial gaits of a simulated salamander. *Biological Cybernetics 84*(5), 331–348.

Ijspeert, A. J. and M. Arbib (2000). Visual tracking in simulated salamander locomotion. In J.-A. Meyer, A. Berthoz, D. Floreano, H. Roitblat, and S. Wilson (Eds.), *From Animals to Animats 6: Proceedings of the Sixth International Conference on the Simulation of Adaptive Behavior (SAB2000)*, pp. 88–97. Cambridge, MA: MIT Press.

Ijspeert, A. J., J. Hallam, and D. Willshaw (1998). From lampreys to salamanders: Evolving neural controllers for swimming and walking. In R. Pfeifer, B. Blumberg, J.-A. Meyer, and S. Wilson (Eds.), *From Animals to Animats 5: Proceedings of the Fifth International Conference on the Simulation of Adaptive Behavior (SAB98)*, pp. 390–399. Cambridge, MA: MIT Press.

Ijspeert, A. J., J. Hallam, and D. Willshaw (1999). Evolving swimming controllers for a simulated lamprey with inspiration from neurobiology. *Adaptive Behavior 7*(2), 151–172.

Ijspeert, A. J. and J. Kodjabachian (1999). Evolution and development of a central pattern generator for the swimming of a lamprey. *Artificial Life 5*(3), 247–269.

Jakobi, N. (1997a). Evolutionary robotics and the radical envelope of noise hypothesis. *Adaptive Behavior 6*(2), 131–174.

Jakobi, N. (1997b). Half-baked, ad-hoc and noisy: Minimal simulations for evolutionary robotics. In P. Husbands and I. Harvey (Eds.), *Advances in Artificial Life: Proceedings of the 4th European Conference on Artificial Life (ECAL97)*, pp. 348–357. Cambridge, MA: MIT Press.

Jakobi, N. (1998). Running across the reality gap: Octopod locomotion evolved in a minimal simulation. In P. Husbands and J.-A. Meyer (Eds.), *Proceedings*

of the First European Workshop on Evolutionary Robotics, pp. 39–58. Berlin: Springer-Verlag.

Jakobi, N., P. Husbands, and I. Harvey (1995). Noise and the reality gap: The use of simulation in evolutionary robotics. In F. Moran, A. Moreno, J. Merelo, and P. Chancon (Eds.), *Advances in Artificial Life: Proceedings of the 3rd European Conference in Artificial Life (ECAL95)*, pp. 704–720. Berlin: Springer-Verlag.

KanGAL (2003). NSGA-II source code (real and binary-coded + constraint handling) [online]. http://www.iitk.ac.in/kangal/code/nsga2code.tar [cited - 11 January 2003].

Kauffman, S. A. (1993). *The Origins of Order*. New York: Oxford University Press.

Kelly, G. A. (1955). *The Psychology of Personal Constructs*. New York: Norton.

Keymeulen, D., M. Durantez, K. Konaka, Y. Kuniyoshi, and T. Higuchi (1996). An evolutionary robot navigation system using a gate-level evolvable hardware. In T. Higuchi, M. Iwata, and W. Liu (Eds.), *Proceedings of the First International Conference on Evolvable Systems: From Biology to Hardware*, pp. 195–209. Berlin: Springer-Verlag.

Keymeulen, D., M. Iwata, K. Konaka, R. Suzuki, Y. Kuniyoshi, and T. Higuchi (1998). Off-line model-free and on-line model-based evolution for tracking navigation using evolvable hardware. In P. Husbands and J.-A. Meyer (Eds.), *Proceedings of the First European Workshop on Evolutionary Robotics*, pp. 211–226. Berlin: Springer-Verlag.

Kim, D.-E. and J. Hallam (2002). An evolutionary approach to quantify internal states needed for the Woods problem. In B. Hallam, D. Floreano, G. Hayes, J. Hallam, and J.-A. Meyer (Eds.), *From Animals to Animats 7: Proceedings of the Seventh International Conference on the Simulation of Adaptive Behavior (SAB2002)*, pp. 312–322. Cambridge, MA: MIT Press.

Kitano, H. (1990). Designing neural networks using genetic algorithms with graph generation system. *Complex Systems 4*(4), 461–476.

Knowles, J. D. and D. Corne (2000). Approximating the nondominated front using the Pareto archived evolution strategy. *Evolutionary Computation 8*(2), 149–172.

Kodjabachian, J. and J.-A. Meyer (1998a). Evolution and development of modular control architectures for 1-D locomotion in six-legged animats. *Connection Science 10*, 211–237.

Kodjabachian, J. and J.-A. Meyer (1998b). Evolution and development of neural controllers for locomotion, gradient-following, and obstacle-avoidance in artificial insects. *IEEE Transactions on Neural Networks 9*, 796–812.

Kolmogorov, A. N. (1965). Three approaches to the quantitative definition of information. *Problems of Information Transmission 1*, 1–7.

Komosinski, M. (2000). The world of Framsticks: Simulation, evolution, interaction. In J.-C. Heudin (Ed.), *Proceedings of the 2nd International Conference on Virtual Worlds (VW 2000)*, pp. 214–224. Berlin: Springer-Verlag.

Komosinski, M., G. Koczyk, and M. Kubiak (2001). On estimating similarity of artificial and real organisms. *Theory in Biosciences 120*(3–4), 271–286.

Komosinski, M. and M. Kubiak (2001). Taxonomy in Alife: Measures of similarity for complex artificial organisms. In J. Keleman and P. Sosik (Eds.), *Advances in Artificial Life: Proceedings of the 6th European Conference on Artificial Life (ECAL01)*, pp. 685–694. Berlin: Springer-Verlag.

Komosinski, M. and A. Rotaru-Varga (2000). From directed to open-ended evolution in a complex simulation model. In M. Bedau, J. McCaskill, N. Packard, and S. Rasmussen (Eds.), *Artificial Life VII: Proceedings of the Seventh International Conference on the Simulation and Synthesis of Living Systems*, pp. 293–299. Cambridge, MA: MIT Press.

Komosinski, M. and A. Rotaru-Varga (2001). Comparison of different genotype

encodings for simulated three-dimensional agents. *Artificial Life 7*(4), 395–418.

Komosinski, M. and S. Ulatowski (1999). Framsticks: Towards a simulation of a nature-like world, creatures and evolution. In D. Floreano, J.-D. Nicoud, and F. Mondada (Eds.), *Advances in Artificial Life: Proceedings of the 5th European Conference on Artificial Life (ECAL99)*, pp. 261–265. Berlin: Springer-Verlag.

Koza, J. R. and J. P. Rice (1991). Genetic generation of both the weights and architecture for a neural network. In *Proceedings of the 1991 International Joint Conference on Neural Networks*, Volume 2, pp. 397–404. Piscataway, NJ: IEEE Press.

Kumar, R. and P. Rockett (2002). Improved sampling of the Pareto-front in multiobjective genetic optimizations by steady-state evolution: A Pareto converging genetic algorithm. *Evolutionary Computation 10*(3), 283–314.

Lara, B., M. Hulse, and F. Pasemann (2001). Evolving neuro-modules and the interfaces to control autonomous robots [online]. In *Proceedings of the World Multiconference on Systemics, Cybernetics and Informatics (SCI 2001)*, Volume 9, Orlando, US, pp. 259–264. http://citeseer.nj.nec.com/lara01evolving.html [cited - 4 January 2003].

Laumanns, M., L. Thiele, K. Deb, and E. Zitzler (2002). Combining convergence and diversity in evolutionary multiobjective optimization. *Evolutionary Computation 10*(3), 263–282.

Lee, W.-P., J. Hallam, and H. J. Lund (1996). A hybrid GP/GA approach for co-evolving controllers and robot bodies to achieve fitness-specific tasks. In *Proceedings of the 3rd IEEE International Conference on Evolutionary Computation*, pp. 384–389. Piscataway, NJ: IEEE Press.

Leger, P. C. (1999). *Automated Synthesis and Optimization of Robot Configurations: An Evolutionary Approach*. Unpublished PhD thesis, Carnegie Mellon University, Pennsylvania.

Lewin, R. (1993). *Complexity: Life at the Edge of Chaos*. London: Phoenix.

Lichtensteiger, L. and P. Eggenberger (1999). Evolving the morphology of a compound eye on a robot. In *Proceedings of the Third European Workshop on Advanced Mobile Robots (Eurobot '99)*, pp. 127–134. Piscataway, NJ: IEEE Press.

Lipsitch, M. (1991). Adaptation on rugged landscapes generated by iterated local interactions of neighboring genes. In R. K. Belew and L. B. Booker (Eds.), *Proceedings of the Fourth International Conference on Genetic Algorithms*, pp. 128–135. San Mateo, CA: Morgan Kaufmann.

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

Lund, H. H. (2001). Adaptive robotics in entertainment. *Applied Soft Computing 1*, 3–20.

Lund, H. H. and J. Hallam (1997). Evolving sufficient robot controllers. In *Proceedings of the 4th IEEE International Conference on Evolutionary Computation*, pp. 495–499. Piscataway, NJ: IEEE Press.

Lund, H. H., J. Hallam, and W.-P. Lee (1997). Evolving robot morphology. In *Proceedings of the 4th IEEE International Conference on Evolutionary Computation*, pp. 197–202. Piscataway, NJ: IEEE Press.

Lyon, M. L. (1993). Complexity and emergence: The seduction and reduction of non-linear models in the social sciences. In D. Green and T. Bossomaier (Eds.), *Complex Systems: From Biology to Computation*, pp. 173–180. Amsterdam: IOS Press.

Macki, J. and A. Strauss (1982). *Introduction to Optimal Control Theory*. New York: Springer-Verlag.

Mainzer, K. (1997). *Thinking in Complexity: The Complex Dynamics of Matter, Mind and Mankind* (3rd ed.). Berlin: Springer-Verlag.

Manderick, B., M. de Weger, and P. Spiessens (1991). The genetic algorithm and

the structure of the fitness landscape. In R. K. Belew and L. B. Booker (Eds.), *Proceedings of the Fourth International Conference on Genetic Algorithms*, pp. 143–150. San Mateo, CA: Morgan Kaufmann.

Mandik, P. (2002). Synthetic neuroethology. *Metaphilosophy 33*(1–2), 11–29.

Mataric, M. (1997). Studying the role of embodiment in cognition. *Cybernetics and Systems 28*(6), 457–470.

Mataric, M. and D. Cliff (1996). Challenges in evolving controllers for physical robots. *Robotics and Autonomous Systems 19*, 67–83.

Maynard Smith, J. and E. Szathmary (1995). *The Major Transitions in Evolution*. New York: W.H. Freeman and Company.

McGeer, T. (1990). Passive dynamic walking. *International Journal of Robotics Research 9*(2), 62–82.

Menczer, F., M. Degeratu, and W. N. Street (2000). Efficient and scalable Pareto optimization by evolutionary local selection algorithms. *Evolutionary Computation 8*(2), 223–247.

Migliono, O. and R. Walker (2002). Genetic redundancy in evolving populations of simulated robots. *Artificial Life 8*(3), 265–277.

Miller, G. F., P. M. Todd, and S. U. Hegde (1989). Designing neural networks using genetic algorithms. In J. Schaffer (Ed.), *Proceedings of the Third International Conference on Genetic Algorithms*, pp. 379–384. San Mateo, CA: Morgan Kaufmann.

Nehaniv, C. L. (2000a). Measuring evolvability as the rate of complexity increase. *Artificial Life 6*(1), 45–67.

Nehaniv, C. L. (2000b). Measuring evolvability as the rate of complexity increase [online]. In C. Maley and E. Boudreau (Eds.), *Artificial Life 7 Workshop Proceedings*, Portland, US, pp. 55–57. http://homepage.feis.herts.ac.uk/∼nehaniv/ [cited - 30 August 2002].

Neyman, A. and D. Okada (1999). Strategic entropy and complexity in repeated games. *Games and Economic Behavior 29*(1–2), 191–223.

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

Nolfi, S. (2002). Power and limits of reactive agents. *Neurocomputing 42*, 119–145.

Nolfi, S. and D. Floreano (1999). Learning and evolution. *Autonomous Robots 7*(1), 89–113.

Nolfi, S. and D. Floreano (2000). *Evolutionary Robotics: The Biology, Intelligence, and Technology of Self-Organizing Machines*. Cambridge, MA: MIT Press/Bradford Books.

Nolfi, S. and D. Floreano (2002). Synthesis of autonomous robots through evolution. *Trends in Cognitive Science 6*(1), 31–36.

Nordin, P. and W. Banzhaf (1996). An on-line method to evolve behavior and to control a miniature robot in real time with genetic programming. *Adaptive Behavior 5*(2), 107–140.

Oliveira, G. M., J. C. Bortot, and P. P. de Oliveira (2002). Multiobjective evolutionary for one-dimensional cellular automata in the density classification task. In R. Standish, M. Bedau, and H. Abbass (Eds.), *Artificial Life VIII: The 8th International Conference on Artificial Life*, pp. 202–206. Cambridge, MA: MIT Press.

Oliveira, G. M., P. P. de Oliveira, and N. Omar (2000). Evolving solutions of the density classification task in 1D cellular automata, guided by parameters that estimate their behavior. In M. Bedau, J. McCaskill, N. Packard, and S. Rasmussen (Eds.), *Artificial Life VII: Proceedings of the Seventh International Conference on the Simulation and Synthesis of Living Systems*, pp. 426–436. Cambridge, MA: MIT Press.

Otsu, K., A. Ishiguro, A. Fujii, T. Aoki, and P. Eggenberger (2001). Evolving an adaptive controller for a quadruped-robot with dynamically-rearranging neural networks. In *Proceedings of the IEEE/RSJ International Conference*

on Intelligent Robots and Systems (IROS2001), Volume 4, pp. 2036–2044. Piscataway, NJ: IEEE Press.

Pachepsky, E., T. Taylor, and S. Jones (2002). Mutualism promotes diversity and stability in a simple artificial ecosystem. *Artificial Life 8*(1), 5–24.

Paredis, J. (1995). The symbiotic evolution of solutions and their representation. In L. Eshelman (Ed.), *Proceedings of the Sixth International Conference on Genetic Algorithms*, pp. 359–365. San Mateo, CA: Morgan Kaufmann.

Parmee, I. C., D. Cvetkovic, A. H. Watson, and C. R. Bonham (2000). Multi-objective satisfaction within an interactive evolutionary design environment. *Evolutionary Computation 8*(2), 197–222.

Pasemann, F., U. Steinmetz, M. Hulse, and B. Lara (2001a). Evolving brain structure for robot control. In J. Mira and A. Prieto (Eds.), *Proceedings of the International Work-Conference on Artificial and Natural Neural Networks (IWANN'2001)*, Volume 2, pp. 410–417. Berlin: Springer-Verlag.

Pasemann, F., U. Steinmetz, M. Hulse, and B. Lara (2001b). Robot control and the evolution of modular neurodynamics. *Theory in Biosciences 120*(3–4), 311–326.

Paul, C. and J. C. Bongard (2001). The road less travelled: Morphology in the optimization of biped robot locomotion. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS2001)*, Volume 1, pp. 226–232. Piscataway, NJ: IEEE Press.

Pfeifer, R. (2000). On the role of morphology and materials in adaptive behavior. In J.-A. Meyer, A. Berthoz, D. Floreano, H. Roitblat, and S. Wilson (Eds.), *From Animals to Animats 6: Proceedings of the Sixth International Conference on the Simulation of Adaptive Behavior (SAB2000)*, pp. 23–32. Cambridge, MA: MIT Press.

Pfeifer, R. and C. Scheier (1999). *Understanding Intelligence*. Cambridge, MA: MIT Press.

Philippides, A., P. Husbands, T. Smith, and M. O'Shea (2002). Fast and loose: Biologically inspired couplings. In R. Standish, M. Bedau, and H. Abbass (Eds.), *Artificial Life VIII: The 8th International Conference on Artificial Life*, pp. 293–301. Cambridge, MA: MIT Press.

Pirjanian, P. (1998). Multiple objective action selection in behavior-based control [online]. In *Proceedings of the 6th Symposium for Intelligent Robotic Systems*, Edinburgh, UK, pp. 83–92. http://robotics.jpl.nasa.gov/people/paolop/publications/index.html [cited - 8 January 2003].

Pirjanian, P. (2000). Multiple objective behavior-based control. *Robotics and Autonomous Systems 31*(1-2), 53–60.

Pollack, J. B., H. Lipson, S. G. Ficici, P. Funes, and G. S. Hornby (2002). Evolutionary techniques in physical robotics. In P. J. Bentley and D. W. Corne (Eds.), *Creative Evolutionary Systems*, Chapter 21, pp. 511–523. San Francisco: Morgan Kaufmann.

Pollack, J. B., H. Lipson, G. S. Hornby, and P. Funes (2001). Three generations of automatically designed robots. *Artificial Life 7*(3), 215–223.

Potter, M. A. and K. A. De Jong (2000). Cooperative coevolution: An architecture for evolving coadapted subcomponents. *Evolutionary Computation 8*(1), 1–29.

Ray, T. S. (2000). Aesthetically evolved virtual pets [online]. In C. Maley and E. Boudreau (Eds.), *Artificial Life 7 Workshop Proceedings*, Portland, US, pp. 158–161. http://www.hip.atr.co.jp/∼ray/pubs/alife7a/ [cited - 27 February 2002].

Reeve, R. (1999). *Generating Walking Behaviors in Legged Robots*. Unpublished PhD thesis, University of Edinburgh, Scotland.

Reil, T. and P. Husbands (2002). Evolution of central pattern generators for bipedal walking in a real-time physics environment. *IEEE Transactions on Evolutionary Computation 6*(2), 159–168.

Reil, T. and C. Massey (2001). Biologically inspired control of physically simulated bipeds. *Theory in Biosciences 120*(3–4), 327–339.

Ronald, E. M. and M. Sipper (2001). Surprise versus unsurprise: Implications of emergence in robotics. *Robotics and Autonomous Systems 37*, 19–24.

Rosin, C. D. and R. K. Belew (1997). New methods for competitive coevolution. *Evolutionary Computation 5*(1), 1–29.

Rumelhart, D. E., G. E. Hinton, and R. J. Williams (1986). Learning internal representations by error propagation. In D. Rumelhart and J. McClelland (Eds.), *Parallel Distributed Processing*, Volume 1, pp. 381–362. Cambridge, MA: MIT Press.

Runyon, R. P., A. Haber, D. J. Pittenger, and K. A. Coleman (1996). *Fundamentals of Behavioral Statistics* (8th ed.). Boston, MA: McGraw-Hill.

Schaffer, J. D. (1984). *Some Experiments in Machine Learning Using Vector Evaluated Genetic Algorithms*. Unpublished PhD thesis, Vanderbilt University, Tennessee.

Shackleton, M., R. Shipman, and M. Ebner (2000). An investigation of redundant genotype-phenotype mappings and their role in evolutionary search. In *Proceedings of the 2000 Congress on Evolutionary Computation (CEC2000)*, Volume 1, pp. 493–500. Piscataway, NJ: IEEE Press.

Shalizi, C. R. (2001). *Causal Architecture, Complexity and Self-Organization in Time Series and Cellular Automata*. Unpublished PhD thesis, University of Wisconsin at Madison, Wisconsin.

Shannon, C. E. (1948). A mathematical theory of communication. *The Bell System Technical Journal 27*(3), 379–423.

Sims, K. (1994a). Evolving 3D morphology and behavior by competition. In R. Brooks and P. Maes (Eds.), *Artificial Life IV: Proceedings of the Fourth International Workshop on the Synthesis and Simulation of Living Systems*, pp. 28–39. Cambridge, MA: MIT Press.

Sims, K. (1994b). Evolving virtual creatures. In A. Glassner (Ed.), *Proceedings of the 21st Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '94)*, pp. 15–22. New York: ACM Press.

Smith, C. U. (1994). The complexity of brains: A biologist's view. In R. Stonier and X. Yu (Eds.), *Complex Systems: Mechanisms of Adaptation*, pp. 93–100. Amsterdam: IOS Press.

Smith, T., P. Husbands, P. Layzell, and M. O'Shea (2002). Fitness landscapes and evolvability. *Evolutionary Computation 10*(1), 1–34.

Smith, T., P. Husbands, and M. O'Shea (2001a). Neutral networks in an evolutionary robotics search space. In *Proceedings of the 2001 Congress on Evolutionary Computation (CEC2001)*, Volume 1, pp. 136–145. Piscataway, NJ: IEEE Press.

Smith, T., P. Husbands, and M. O'Shea (2001b). Not measuring evolvability: Initial investigation of an evolutionary robotics search space. In *Proceedings of the 2001 Congress on Evolutionary Computation (CEC2001)*, Volume 1, pp. 9–16. Piscataway, NJ: IEEE Press.

Smith, T., P. Husbands, A. Philippides, and M. O'Shea (2002). Temporally adaptive networks: Analysis of GasNet robot control networks. In R. Standish, M. Bedau, and H. Abbass (Eds.), *Artificial Life VIII: The 8th International Conference on Artificial Life*, pp. 274–282. Cambridge, MA: MIT Press.

Smith, T., A. Philippides, P. Husbands, and M. O'Shea (2002). Neutrality and ruggedness in robot landscapes. In *Proceedings of the 2002 Congress on Evolutionary Computation (CEC2002)*, Volume 2, pp. 1348–1353. Piscataway, NJ: IEEE Press.

Spronck, P., I. G. Sprinkhuizen-Kuyper, and E. O. Postma (2001). Evolutionary learning of a neural robot controller. In M. Mohammadian (Ed.), *2001 International Conference on Computational Intelligence for Modelling, Control and Automation (CIMCA'2001)*, Las Vegas, USA, pp. 511–519. ISBN 0-858-89847-0 [CD-ROM].

Srinivas, N. and K. Deb (1994). Multi-objective function optimization using non-dominated sorting genetic algorithm. *Evolutionary Computation 2*(3), 221–248.

Standish, R. K. (2001). On complexity and emergence [online]. *Complexity International 9*. http://life.csu.edu.au/ci/vol09/standi09/ [cited - 30 August 2002].

Storn, R. and K. Price (1995). Differential evolution: A simple and efficient adaptive scheme for global optimization over continuous spaces. Technical Report TR-95-012, International Computer Science Institute, Berkeley.

Szathmary, E., F. Jordan, and C. Pal (2001). Can genes explain biological complexity? *Science 292*, 1315–1316.

Tan, K. C., T. H. Lee, and E. F. Khor (1999). Control system design automation with robust tracking thumbprint performance using a multi-objective evolutionary algorithm [online]. In *Proceedings of the 1999 International Symposium on Computer Aided Control System Design*, Hawaii, USA, pp. 498–503. http://www.lania.mx/ ccoello/EMOO/EMOObib.html#T [cited - 30 November 2002].

Tan, K. C. and Y. Li (1997). Multi-objective genetic algorithm based time and frequency domain design unification of linear control systems [online]. In *Proceedings of the IFAC/IEEE International Symposium on Artificial Intelligence in Real-Time Control*, Kuala Lumpur, Malaysia, pp. 61–66. http://www.mech.gla.ac.uk/Research/Control/Publications/Rabstracts/abs97007.html [cited - 30 November 2002].

Taylor, T. (2000). Artificial life techniques for generating controllers for physically modelled characters [online]. In Q. Mehdi and N. Gough (Eds.), *Proceedings of the First International Conference on Intelligent Games and Simulation (GAME-ON 2000)*, London, UK. http://citeseer.nj.nec.com/taylor00artificial.html [cited - 30 December 2002].

Taylor, T. (2002). Evolution of morphology and behavior for physically modelled creatures [online]. http://www.dai.ed.ac.uk/homes/timt/research/embody.html [cited - 5 September 2002].

Taylor, T. and C. Massey (2001). Recent developments in the evolution of morphologies and controllers for physically simulated creatures. *Artificial*

Life 7(1), 77–87.

Teo, J. and H. A. Abbass (2002a). Coordination and synchronization of locomotion in a virtual robot. In L. Wang, J. Rajapakse, K. Fukushima, S. Lee, and X. Yao (Eds.), *Proceedings of the 9th International Conference on Neural Information Processing (ICONIP'02)*, Volume 4, Singapore, pp. 1931–1935. Nanyang Technological University: ISBN 981-04-7525-X.

Teo, J. and H. A. Abbass (2002b). Multi-objectivity for brain-behavior evolution of a physically-embodied organism. In R. Standish, M. Bedau, and H. Abbass (Eds.), *Artificial Life VIII: The 8th International Conference on Artificial Life*, pp. 312–318. Cambridge, MA: MIT Press.

Teo, J. and H. A. Abbass (2002c). Trading-off mind complexity and locomotion in a physically simulated quadruped. In L. Wang, K. Tan, T. Furuhashi, J. Kim, and X. Yao (Eds.), *Proceedings of the 4th Asia-Pacific Conference on Simulated Evolution And Learning (SEAL'02)*, Volume 2, Singapore, pp. 776–780. Nanyang Technological University: ISBN 981-04-7523-3.

Teo, J. and H. A. Abbass (2003). Search space difficulty of evolutionary neuro-controlled legged robots. *International Journal of Knowledge-Based Intelligent Engineering Systems (Accepted to appear)*.

Teo, J., M. H. Nguyen, and H. A. Abbass (2003). Multi-objectivity as a tool for constructing hierarchical complexity. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2003) (Accepted to appear)*, Lecture Notes in Computer Science (LNCS), Chicago, IL. Springer-Verlag.

Thompson, A. (1995). Evolving electronic robot controllers that exploit hardware resources. In F. Moran, A. Moreno, J. Merelo, and P. Chacon (Eds.), *Advances in Artificial Life: Proceedings of the Third European Conference in Artificial Life (ECAL95)*, pp. 640–656. Berlin: Springer-Verlag.

Thompson, A. (1997). Artificial evolution in the physical world. In T. Gomi (Ed.), *Proceedings of the 5th International Symposium on Evolutionary Robotics: From Intelligent Robots to Artificial Life (ER'97)*, pp. 101–125. Ontario: AAI

Books.

Turing, A. M. (1936). On computable numbers, with an application to the entscheidungsproblem. *Proceedings of the London Mathematical Society (Series 2) 42*, 230–265.

Van Veldhuizen, D. A. and G. B. Lamont (2000a). Multiobjective evolutionary algorithms: Analyzing the state-of-the-art. *Evolutionary Computation 8*(2), 125–147.

Van Veldhuizen, D. A. and G. B. Lamont (2000b). Multiobjective optimization with messy genetic algorithms. In *Proceedings of the 2000 ACM Symposium on Applied Computing*, pp. 470–476. New York: ACM Press.

Vapnik, V. N. and A. Y. Chervonenkis (1971). On the uniform convergence of relative frequencies of events to their probabilities. *Theoretical Probability and Its Applications 16*(2), 264–280.

Varela, F. J. (1995). The re-enchantment of the concrete. In L. Steels and R. Brooks (Eds.), *The Artificial Life Route to Artificial Intelligence: Building Embodied, Situated Agents*, Chapter 1, pp. 11–22. Hillsdale, NJ: Lawrence Erlbaum Associates Publishers.

Vassilev, V. K., T. C. Fogarty, and J. F. Miller (2000). Information characteristics and the structure of landscapes. *Evolutionary Computation 8*(1), 31–60.

Vassilev, V. K. and J. F. Miller (2000). The advantages of landscape neutrality in digital circuit evolution. In *Proceedings of the Third International Conference on Evolvable Systems: From Biology to Hardware (ICES'2000)*, pp. 252–263. Berlin: Springer-Verlag.

Waldrop, M. M. (1994). *Complexity: The Emerging Science at the Edge of Order and Chaos*. London: Penguin Books.

Warren, T. and E. Gibson (2002). The influence of referential processing on sentence complexity. *Cognition 85*(1), 79–112.

Watson, R. A., S. G. Ficici, and J. B. Pollack (2002). Embodied evolution: Dis-

tributing an evolutionary algorithm in a population of robots. *Robotics and Autonomous Systems 39*, 1–18.

Weinberger, E. D. (1990). Correlated and uncorrelated fitness landscapes and how to tell the difference. *Biological Cybernetics 63*, 325–336.

Wolfram, S. (2002). *A New Kind of Science*. Champaign, IL: Wolfram Media Inc.

Wolpert, D. H. and W. G. MacReady (1997). Self-dissimilarity: An empirical measure of complexity. Working Paper 97-12-087, Sante Fe Institute.

Wright, S. (1932). The roles of mutation, inbreeding, crossbreeding and selection in evolution. In D. Jones (Ed.), *Proceedings of the Sixth International Conference on Genetics*, Volume 1, Brooklyn, NY, pp. 356–366.

Wuensche, A. (1999). Classifying cellular automata automatically: Finding gliders, filtering, and relating space-time patterns, attractor basins, and the Z parameter. *Complexity 4*(3), 47–66.

Yao, X. (1999). Evolving artificial neural networks. *Proceedings of the IEEE 87*(9), 1426–1447.

Zitzler, E. (1999). *Evolutionary Algorithms for Multiobjective Optimization: Methods and Applications*. PhD thesis, Swiss Federal Institute of Technology (ETH), Zurich, Switzerland.

Zitzler, E. (2002). Evolutionary algorithms for multiobjective optimization. In K. Giannakoglou, D. Tsahalis, J. Periaux, and T. Fogarty (Eds.), *Evolutionary Methods for Design, Optimization and Control*, pp. 19–26. Barcelona, Spain: International Center for Numerical Methods in Engineering (CIMNE).

Zitzler, E., K. Deb, and L. Thiele (2000). Comparison of multiobjective evolutionary algorithms: Empirical results. *Evolutionary Computation 8*(2), 173–195.

Zitzler, E., M. Laumanns, and L. Thiele (2001). SPEA2: Improving the strength Pareto evolutionary algorithm. Technical Report 103, Computer Engineering and Networks Laboratories (TIK), Swiss Federal Institute of Technology (ETH), Zurich, Switzerland.

Zitzler, E. and L. Thiele (1999). Multiobjective evolutionary algorithms: A comparative case study and the strength Pareto approach. *IEEE Transactions on Evolutionary Computation 3*(4), 257–271.

Zydallis, J. B., D. A. Van Veldhuizen, and G. B. Lamont (2001). A statistical comparison of multiobjective evolutionary algorithms including the MOMGA-II. In E. Zitzler, K. Deb, L. Thiele, C. Coello Coello, and D. Corne (Eds.), *Proceedings of the First International Conference on Evolutionary Multi-Criterion Optimization (EMO'01)*, pp. 226–240. Berlin: Springer-Verlag.

Appendix A

Contents of the Accompanying CD-ROM

The CD-ROM accompanying this thesis contains videos of the artificial creature visualized in simulation and additional graphs that were not included in the thesis. The directory structure of the CD-ROM's contents is shown in Figure A. On machines with autorun features enabled, the CD-ROM should play automatically. Otherwise, it is recommended to begin using the CD-ROM by first reading the `readme.txt` file, and then pointing your web browser to `index.html`. All videos and graphs on the CD-ROM are accessible through this index file.

The graphs require Adobe Acrobat Reader to view. A link to download a copy can be found in the `index.html` file. To view the graphs, simply click on the corresponding links that appear on the relevant pages. The videos require an MPEG viewer. If using Internet Explorer 5.5 (or higher) or Netscape 6.2 (or higher) on a personal computer running Microsoft Windows 98 (or higher), these videos will play automatically by clicking on the corresponding links. Otherwise, a link to on-line instructions on how to install the appropriate MPEG players can be found in the `index.html` file.

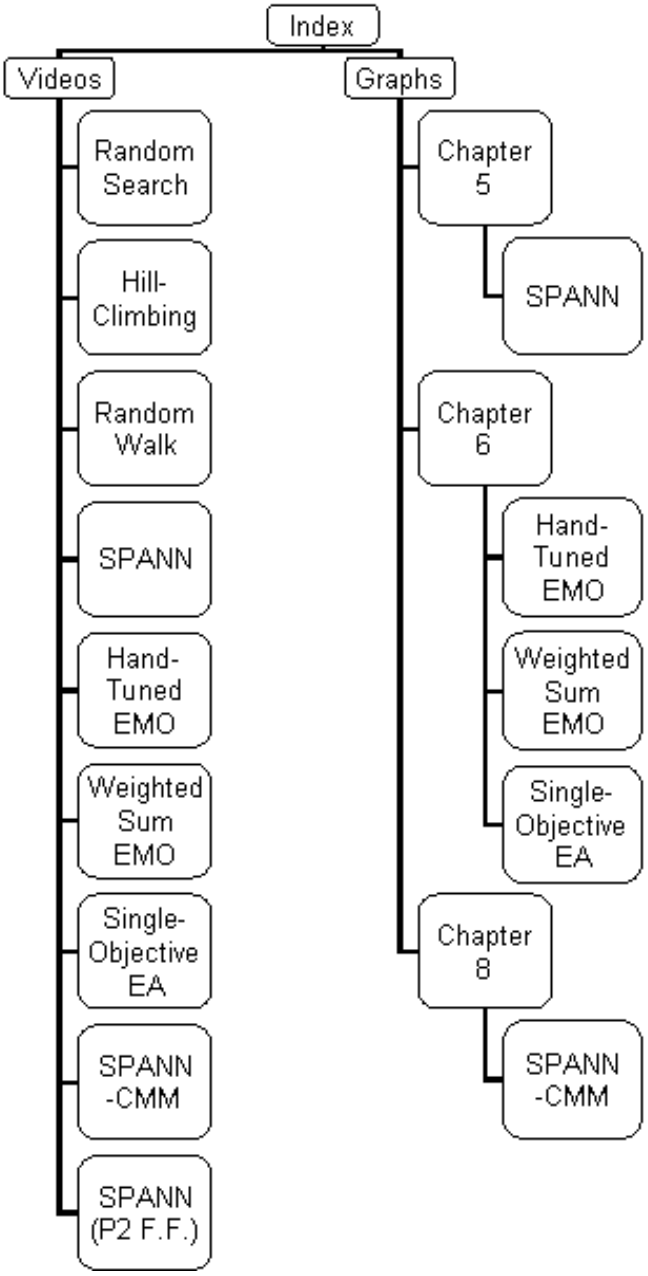


Figure A.1: Index to the contents of the accompanying CD-ROM. The last video listed as "SPANN (P2 F.F.)" refers to the locomotion behavior evolved using the P2 fitness function explained in Section 7.4.4. Videos for the best SPANN controller operating with the presence of noise can be found under the `Videos\SPANN` sub-directory.