Design, Evaluation and Comparison of Evolution and Reinforcement Learning Models.

Clinton Brett Mclean

Submitted in fulfillment of the requirements
for the degree of Master of Economics,
in the Department of Computer Science,
Rhodes University.

April 2001

Abstract

This work presents the design, evaluation and comparison of evolution and reinforcement learning models, in isolation and combined in Darwinian and Lamarckian frameworks, with a particular emphasis being placed on their adaptive nature in response to environments that become increasingly unstable. Our ultimate objective is to determine whether hybrid models of evolution and learning can demonstrate adaptive qualities beyond those of such models when applied in isolation.

This work demonstrates the limitations of evolution, reinforcement learning and Lamarckian models in dealing with increasingly unstable environments, while noting the effective adaptive nature of a Darwinian model to assimilate increasing levels of instability. This is shown to be a result of the Darwinian evolution model's ability to separate learning at two levels, the population's experience of the environment over the course of many generations and the individual's experience of the environment over the course of its lifetime. Thus, knowledge relating to the general characteristics of the environment over many generations can be maintained in the population's genotypes with phenotype (reinforcement) learning being utilized to adapt a particular agent to the particular characteristics of its environment.

Lamarckian evolution, though, is shown to demonstrate adaptive characteristics that are highly effective in response to the stable environments. Selection and reproduction combined with reinforcement learning creates a model that has the ability to utilize useful knowledge produced by reinforcements, as opposed to random mutations, to accelerate the search process. As a result the influence of individual learning on the populations evolution is shown to be more successful when applied in the more direct Lamarckian form.

Based on our results demonstrating the success of Lamarckian strategies in stable environments and Darwinian strategies in unstable environments, hybrid Darwinian/Lamarckian models are created with a view towards combining the advantages of both forms of evolution to produce a superior adaptive capability.

Our investigation demonstrates that such hybrid models can effectively combine the adaptive advantageous of both Darwinian and Lamarckian evolution to provide a more effective capability of adapting to a range of conditions, from stable to unstable, appropriately adjusting the required degree of inheritance in response to the requirements of the environment.

Acknowledgements

The supervision of Professor Shaun Bangay was instrumental to the development of this research project and the quality of the ultimate thesis. Thanks also to Professor Peter Clayton who provided an informative commentary on an earlier version of the work. Finally, Professor Patrick Terry's proof-reading skills were greatly appreciated.

Contents

Chapter 1: Introduction

The nature of this work is an investigation of the adaptive characteristics of evolution and reinforcement learning models when applied in isolation and combined in Darwinian and Lamarckian frameworks. Five models of evolution and learning are created with the objective of maximizing their adaptive abilities in an environment of changing stability. These models are evaluated with a view towards measuring their effectiveness and extracting their advantages and limitations in dealing with environments that become increasingly unstable.

An important objective is to investigate whether hybrid models of evolution and learning can demonstrate adaptive qualities that such models alone do not possess. How does learning interact with evolution in Darwinian and Lamarckian frameworks and in environments of differing stability and does their combination provide any special adaptive qualities?

Our goals then are to:

- Design several evolution and reinforcement learning models with a view towards maximizing their adaptive abilities in environments of increasing instability.
- Evaluate the effectiveness of the models response to the environment, providing an analysis of their adaptive characteristics.
- Compare the effectiveness of the models in response to various degrees of environmental stability.

This thesis, then, presents the design, evaluation and comparison of evolution and reinforcement learning models with a view towards their adaptability in environments of changing stability and a specific focus on the interaction between learning and evolution in Darwinian and Lamarckian frameworks.

1.1 Background

The following gives a general introduction to evolutionary computation and its relationship with learning, emphasizing the issues that concern us.

Prior to Darwin's theory of evolution, the power responsible for the complex creations in our environment was beyond our understanding and our own capability to reproduce. This is a testament to the power of evolution. Evolutionary computation is the field of Computer Science devoted to studying the use of this power.

Evolution models allow us to solve problems by specifying what is to be achieved rather than how to do it. In nature the ultimate problem is existence. Complexity is shaped and grounded by that natural, ultimate "goal" of survival. Evolutionary computation works by using an ultimate goal to produce goal-satisfying mechanisms. Desired goals can be specified in terms of the criteria to be satisfied.

Evolutionary computation has been applied to a wide variety of problems, including optimization, evolving computer programs, economic models, immune system models, ecological models, social system models, learning and how individual learning and population learning (evolution) interact.

It can be argued that evolutionary computation is applicable to almost every facet of human existence based on the premise that evolution is solely responsible for most, if not all, of the complexity in our experience. Even in the case of complexity not being directly derived from evolutionary mechanisms, it could still be used as a means of reasoning regarding such complexity. For example, evolutionary computation could be used to generate theories that are compatible with pre-established empirical data (i.e., survival of the fittest theory).

In fact, it has been suggested that evolution and the scientific method of conjecture and refutations are analogous to each other [Christianni, 1995], which is inspired by the Evolutionary Epistemology of Donald Campbell and Karl Popper [Campbell, 1980; Popper, 1985]. Essentially what this means is that conjectures (new hypothesis) are equivalent to mutation, crossover, …etc producing new structures, and that refutation is equivalent to selection (or non-selection).

Evolutionary computation, then, is a technique for giving machines the ability to make their own discoveries. However, for it to be a useful problem solving technique, evolution needs to be directed.

Natural evolution is orientated towards the ultimate goal of survival. It discovers structures, which are adapted to their environment. Evolutionary computation can be used to direct evolution by designing an appropriate virtual environment and/or imposing a selection (fitness) function. For example, the design of the environment includes the problem to be solved. Adaptations to the environment, then, are also solutions to the problem. A fitness function can also be used to select appropriate, important attributes of the adaptations as measures of fitness. Those adaptations that meet the criteria to the greatest extent are the fittest solutions.

However, evolution is not the only important adaptive mechanism. The other is learning. They can be related as follows: Evolution is equivalent to adaptation at the population/generation level, while learning provides individuals with the ability to adapt during their lifetime. They are similar processes taking place over different time scales in relation to different, but related feedback.

Evolution or population level learning can adapt to long-term, global statistical regularities in the environment as experienced by the population, whereas individual level learning can adapt to short-term, local statistical regularities as experienced by the individual. As a result combining the two processes can produce individuals with an innate, general knowledge of their world, over time and space, that can be further refined to their particular situation.

For example, if a certain object is normally always edible then evolution will take advantage of this and encode behaviour to eat this object. There is no need for the behaviour to be learnt.

However, if there are other objects in the environment that change from edible to non-edible, depending on the situation, then a learning mechanism is required to take advantage of this regularity. Thus, there is a selective pressure for such a mechanism. Those that can develop the mechanism will take advantage of the extra rewards, since such individuals can adapt to situations that innate, fixed, instinctual behaviour produced by evolution may be unable to (or at least take a long time doing so).

Our research focuses on the interaction between evolution and learning and whether hybrid models can demonstrate adaptive qualities that either method alone does not possess.

At a population level death is a feedback mechanism; incorrect adaptations (individuals) are removed from the population. Death, thus, alters the knowledge encoded in the population. What are needed at an individual level are feedback mechanisms to alter structures within an individual appropriate to its adaptive requirements.

Reinforcement learning is our learning method of choice as it is inherently suited to learning in feedback-impoverished environments. It possesses the ability to solve problems based on reinforcements relating to what is to be achieved rather than how to do it. This is particularly useful in environments where the correct behaviour is unknown, but it is still possible to evaluate states. This is also the case with evolution models and for this reason it can be inferred that a combination of reinforcement learning and evolution models can be utilized to produce powerful hybrids for solving the same types of problems that are normally approached using such models in isolation.

By producing hybrid models of evolution and learning it is possibly for learning to occur at two levels, a population and an individual level. This can give rise to a complementary relationship combining the advantages of each model and demonstrating adaptive qualities that either form of learning alone does not possess.

Our investigation combines learning and evolution in Darwinian and Lamarckian frameworks. Darwinian evolution utilizes learning to alter the knowledge of an individual without altering the information encoded in its genes, although individual learning can indirectly affect the development of genetic knowledge as a result of the Baldwin effect [Baldwin, 1896]. Lamarckian evolution allows an individual's learnt knowledge to be directly encoded into the genotype of its offspring.

We analyse both forms of evolution and learning with regard to environments of differing stability and, in addition, investigate the potential of hybrid Darwinian/Lamarckian evolution models.

1.2 Method

An environment is constructed to investigate the adaptive qualities of our evolution and learning models. It is of an Artificial Life type, consisting of food and poison objects. In order to provide differing degrees of stability, the relationship between an object and its benefits to the agent are altered with a set frequency. That is food becomes poison and poison becomes food after a specified interval. This allows us effectively to test the adaptive qualities of our evolution and learning models in environments of changing stability.

Each model is evaluated in environments that range from highly stable to highly unstable. An overriding criteria of evaluation is the adaptive ability of the model to an initially stable environment that becomes increasingly unstable. Thus, an emphasis is placed on the adaptive abilities of each model towards changes in the stability of the environment.

Effectiveness is measured in terms of agent fitness scores, calculated according to the amount of food they consume less the amount of poison. Agents are allowed to evolve and/or learn over a period of 200 generations with the process being repeated 30 times to produce statistically significant results, that is, results with sufficiently low standard deviations. This provides a sample of a model's effectiveness in an environment of a particular stability. The process is then repeated to evaluate the model's effectiveness in environments of increasing instability.

Agents consist of a neural network, which receives and processes visual input regarding the environment to an output action. The evolution model applies mutations to the weights (or connection strengths) of the neural networks, altering the encoded knowledge. Learning is achieved by a reinforcement component that can also affect the neural network's weights. Positive results of behaviour (i.e., locating food) evaluate to positive reinforcement of that behaviour, and vice-versa for detrimental results (i.e., eating poison).

Thus, the agent's neural networks evolve under the influence of a fitness function, while learning is directed by reinforcement feedback signals.

Our evolution and reinforcement learning models are designed and evaluated in isolation, and then combined in the form of Darwinian, Lamarckian and hybrid

Darwinian/Lamarckian evolution models. The effectiveness and adaptive characteristics of each model are then compared.

The following provides a general background to evolutionary computation, reinforcement learning, neural networks and evolutionary approaches to designing neural networks, followed by related work investigating the interaction between evolution and learning.

Section 1 - Background and Related Work

Chapter 2: Background

2.1 Evolutionary Computation

2.1.1 Description

Natural evolution is fundamentally a process of random mutations and natural selection. In order to understand Evolutionary Computation models it is necessary to understand that:

(1) **Selection** is achieved through a fitness function or endogenous selective forces. A fitness function is designed to bias survival rates of solutions with regard to specified criteria. It provides ratings for individual solutions that can be used to influence their future survival. Following the example of nature, selection can also be endogenous to the environment. In this case the dynamics of the environment itself exerts a selective pressure.
(2) **Solutions** exist as individuals in the environment. The **Population** consists of two or more solutions. The initial population can consist of randomised solutions or even partially complete solutions produced by other methods.
(3) **Variation** in solutions is achieved through the application of operators like mutation and crossover (combining information from parent solutions to produce offspring).
(4) The **Environment** is that which can influence and be influenced by the solutions.

If we think of solutions existing as points in a normally hyper-dimensional solution space, with fitness as an extra dimension, then the search operators, normally mutation, crossover and selection, allow the search points to move in the solution space.

For example, mutation will produce new search points surrounding the original solution. Those points that are fitter will have greater reproductive success, thereby moving the search in the direction that produces more optimal solutions. Obviously, though, search points run the risk of falling into local optima. When this occurs further movement in the surrounding area will result in sub-optimal solutions, even though a better or the best solution (global optimum) still exists. Thus, the search points are stuck in a sub-optimal situation. However, since there is a population of search points other areas of the solution space can be searched. With a diverse, widespread population this may result in better optimums or even the global optimum.

Much work has been done in optimising evolutionary search procedures. For example, the search processes can be given the ability to bias their mutations in a certain direction that have shown success in the past. This can be used to provide

velocity and momentum to the mutations. The bias can be included as a variable in the search processes' definitions that, like any other variable, can be operated on by the automated operators of the evolutionary process. Thus, mutation and crossover rates can be incorporated into the evolutionary process, rather than being chosen intuitively or through tedious empirical experimentation. This provides the evolution model with an additional optimization ability.

2.1.2 Historical Development

Several developments in the 60's and 70's gave rise to what is now known as the field of Evolutionary Computation. These developments were not necessarily related and occurred in different parts of the world.

In 1963 two students at the Technical University of Berlin were faced with the engineering problem of finding the optimal shape of bodies in a flow. Inspired by the example of natural mutations in biological systems, random changes were applied to the parameters defining the shape of a body. This was the beginning of the Evolutionary Strategy [Schwefel, 1981], which can be described as a strategy that models certain aspects of natural evolution (selection and mutation) for parameter optimization problems.

A separate, but similar development, is that of Evolutionary Programming. Beginning in 1960, due to Lawrence J. Fogel, and leading to the landmark publication "Artificial Intelligence Through Simulated Evolution" by Fogel, Owens and Walsh in 1966, it was conceived to evolve finite state machines for the purpose of prediction [Fogel, Owens and Walsh, 1966]. Fogel defined predictive ability as being a prerequisite for intelligent behaviour and described how prediction machines might be evolved - that is, finite state machines that would predict symbols in strings generated from an "environment".

Both early Evolutionary Strategies and Evolutionary Programming rely on mutation and selection as their only operators.

More accurate models of gene related evolution observed in biological systems are provided by Genetic Algorithms. Both crossover (combining information from parent structures to produce the offspring) and mutation are used as genetic operators.

There is also a clear distinction between genotype (genetic encoding of the structure) and phenotype (the resulting structure itself). The genotype is represented as a binary string with the phenotype being the resulting structure (arrays, trees, neural network…etc) after a decoding process. The earliest landmark publication in this area was John Holland's 1975 book "Adaptation in Natural and Artificial Systems" [Holland, 1975].

Although Holland laid some neat theoretical foundations (i.e., Schema Theorem, which relates to the nature of the search process and relies on binary representations of solutions) much contemporary work takes the approach of adapting the solution representation (chromosomes) and operators to the data structures involved, thereby taking advantage of problem dependent information. This is not possible after the conversion to a problem independent solution (binary strings). Thus, instead of

transforming the problem to suit a generic genetic algorithm, the genetic algorithm is transformed to suit the problem. Such a change in perception is described in "Genetic Algorithms + Data Structures = Evolution Programs" by Zbigniew Michalewicz [Michalewicz, 1996].

Although there are many differences between the various fields of Evolutionary Computation, we can remove them and come up with a general description for an evolutionary algorithm.

Solutions are represented as individuals in an environment. They undergo changes, due to mutation and/or crossover. The environment/fitness function creates a selective pressure, which manipulates the direction of changes towards an improved solution to the problem (adaptation to the environment).

2.1.3 Applied Areas

Such algorithms have widespread applications. Some of these include the following areas: Optimization, Automatic Programming, Machine and Robot Learning, Economic Models, Immune System Models, Ecological Models, Population Genetics Models, Interactions between Evolution and Learning and Models of Social systems. Melanie Mitchell and Stephanie Forrest provide many references for the above in their paper "Genetic Algorithms and Artificial Life" [Mitchell and Forrest, 1994].

The Hitchhiker's Guide to Evolutionary Computation lists Computer Scientists, Engineers, Roboticists, Cognitive Scientists, Physicists, Biologists, Chemists and Philosophers as researchers who are interested in Evolutionary Computation [Heitkotter and Beasly, 1998].

Even art is not outside of its domain. A paper by Sims, entitled "Artificial Evolution for Computer Graphics" [Sims, 1991], describes an application that evolves structures, images, textures and animations according to the aesthetic preferences of the user, who becomes the fitness function through his or her interaction with the application.

2.1.4 Advantages of Evolutionary Computation

Advantages outlined by David Fogel in his paper "The Advantages of Evolutionary Computation" [Fogel, 1997] include:

- Conceptual simplicity
- Broad applicability
- Outperform classic methods on real problems
- Potential to use knowledge and hybridise with other methods
- Parallelism
- Robust to dynamic changes
- Capability for self-optimization
- Ability to solve problems that have no known solutions

Also, quoting from Balakrishnan, Honavar *Properties of Genetic Representations of Neural Architectures*, Genetic Algorithms "…offer an attractive approach for

efficiently searching vast, complex and deceptive problem spaces" [Balakrishnan and Honavar, 1995b].

2.2 Reinforcement Learning

2.2.1 Description

Reinforcement learning is a method of gaining knowledge through trial and error interaction with a feedback-impoverished environment. That is, feedback consists of scalar values indicating degrees of correctness, rather than the answer itself (supervised learning). This form of learning is commonly applied to agents that must learn to map input states to actions in order to maximize their positive reinforcements.

The problem is that actions can have far reaching effects beyond the immediate reinforcement (if any) that they receive. How can an agent learn the effect that its current actions will have on its foreseeable future or, put differently, which actions in its past are responsible for the current reinforcement? That is, how can responsibility for delayed reinforcement be assigned to previous actions? This is the problem of temporal credit assignment.

2.2.2 TD(0)

Temporal difference learning is a method for dealing with delayed rewards. Sutton's TD (0) algorithm [Sutton, 1988] can be used to provide state evaluations that reinforcement learning can use to find optimal policies (state-action mappings). Each time a state is visited its valuation is updated according to the immediate real reinforcement received from the environment and the expected value of the following state.

It does this by using the following update rule:

$$V(s) = V(s) + \alpha(r + \gamma V(s+1) - V(s))$$

where V(s) is a valuation of the previous state, α is a learning rate, r is the immediate real reward provided by the environment, and $\gamma V(s+1)$ is the estimated value of the immediately following state resulting from the previous action (discounted by γ).

Thus, when a state is visited its value is updated to move closer to $r + \gamma V(s+1)$, which is a sample of the value of V(s). TD (0), therefore, looks one step ahead when adjusting state evaluations.

2.2.3 TD(λ) - Eligibility Traces

Sutton's more general method TD (λ) [Sutton, 1998] uses the concept of an eligibility trace [Klopf, 1972]. This is a measure of the degree to which a particular state has been visited in the past, thereby giving an indication of its responsibility for the current reinforcement.

Eligibility traces decay over time according to the product of a discount parameter, $\gamma(0 \leq \gamma \leq 1)$, and a decay parameter, $\lambda(0 \leq \lambda \leq 1)$.

A state's conventional accumulating trace can be defined by:

$$e(s)' = \begin{cases} \gamma\lambda(e(s) & \text{if } s \neq s(t) \text{ (the state does not equal the current state)} \\ \gamma\lambda(e(s)+1 & \text{if } s = s(t) \text{ (the state is the current state)} \end{cases}$$

where e(s) is the state's original accumulating eligibility trace, e(s)' is the state's updated accumulating eligibility trace, and γ and λ are discount and decay parameters.

Replacing traces, introduced by Singh and Sutton in "Reinforcement Learning with Replacing Traces"[Singh and Sutton, 1996], reset the trace to a maximum value each time the state is visited.

Replacing traces can be defined by:

$$e(s)' = \begin{cases} \gamma\lambda(e(s) & \text{if } s \neq s(t) \text{ (the state does not equal the current state)} \\ 1 & \text{if } s = s(t) \text{ (the state is the current state)} \end{cases}$$

where e(s) is the state's original replacing eligibility trace, e(s)' is the state's updated replacing eligibility trace, and γ and λ are discount and decay parameters.

Conventional accumulating traces, therefore, implement the credit assignment heuristics of recency and frequency [Sutton 1984], whereas replacing traces only implement recency, since the eligibility trace does not accumulate values for the previous times the state was visited.

The differences are clearly shown in the following diagram (Figure 2-1).

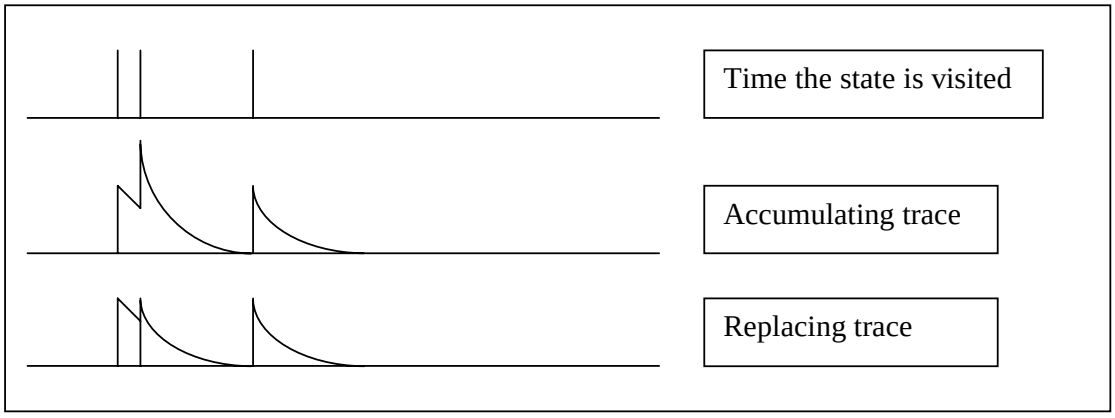


Figure 2-1 Accumulating and replacing eligibility traces.

TD (λ)'s update rule is:

$$V(u) = V(u) + \alpha(r + \gamma V(s+1) - V(s))e(u)$$

where V(u) is the valuation of a particular state (u), α is a learning rate, r is the immediate real reward provided by the environment, γV(s+1) is the estimated value of the immediately following state resulting from the previous action (discounted by γ), and e(u) is the eligibility trace for the state being evaluated.

This update is applied to every state (u) according to its eligibility, e(u).

2.2.4 Adaptive Heuristic Critic

TD algorithms can be used in the form of a strategy called Adaptive Heuristic Critic [Barto, Sutton and Anderson, 1983] to accomplish reinforcement learning.

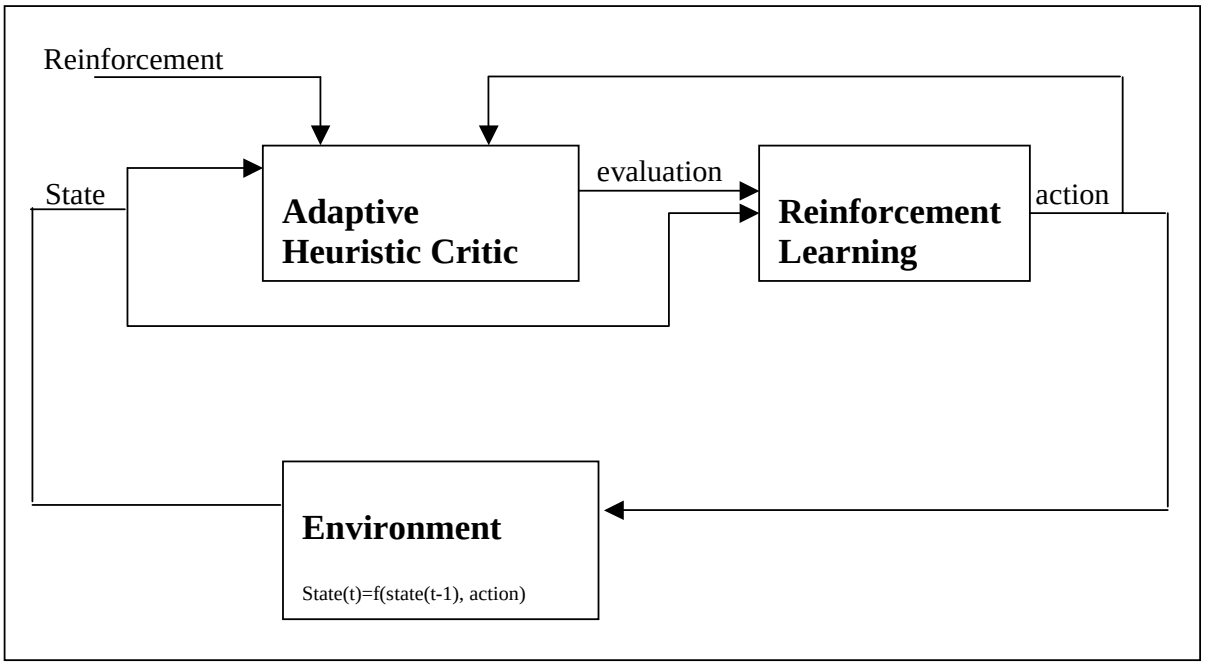


Figure 2-2 Adaptive Heuristic Critic.

The adaptive heuristic critic in the above figure (Figure 2-2) uses a TD algorithm to provide evaluations of policies (state-action mappings) for a reinforcement learning component. Reinforcement learning modifies its policies in order to maximize the heuristic feedback regarding state evaluations as given by the critic. The adaptive heuristic critic, in turn, modifies its evaluations in response to changes in the policy of the reinforcement learning component. In this way TD provides an effective evaluation of states with regard to reinforcements received from the environment that can be utilized by the reinforcement learning component to produce optimal state-action mappings.

In the following section we will describe how reinforcement learning can be used to train neural networks.

2.3 Neural Networks

2.3.1 Description

Neural networks consist of many simple processing units (neurons) that receive and transmit signals to other neurons via weighted connections.

The neurons sum received signals and produce an output signal depending on a neuron activation function, i.e., if the summation exceeds a certain threshold then the neuron fires. The weighted connections encode a neural network's knowledge.

The architecture of a neural network is normally organized in layers. Single-layer networks have one layer of weighted connections separating input neurons from output neurons. Multi-layer networks can be used to solve more complicated problems. Feedforward networks process data in only one direction, from the input neurons to the output neurons. Recurrent networks can feed data back to previous layers.

Our work utilizes neural networks, as their amenability to modification makes them suitable to both evolution and reinforcement learning methods. This means that changes to their structure can be assimilated with a high degree of flexibility. Small changes in their structure normally result in small changes to their behaviour, thereby providing informative feedback as to the appropriate direction to be taken. High-level computer languages are brittle in the sense that small changes to a program's description can result in large changes to the nature of the program's functioning, making evolution and/or learning difficult. In such cases the solution landscape is highly rugged and, thus, difficult to navigate. Neural networks, though, provide a convenient unit of operation to investigate evolution and learning methods.

2.3.2 Neural Network Reinforcement Learning

Reinforcement learning can be used to train neural networks using single feedback values for previous input/output mappings. Positive feedback can be used to alter the network in a manner that increases the probability of an appropriate mapping occurring again and vice-versa for inappropriate mappings. Such reinforcement of input/output pairs is called associative reinforcement learning [Barto and Anandan, 1985].

Ackley and Littman [Ackley and Littman, 1990] extended the backpropagation algorithm [Rumelhart, Hinton and Williams, 1986], used to train neural networks based on desired input/output pairs, to accomplish associative reinforcement learning for multi-layer neural networks. This work demonstrated that the generalizing capabilities of neural networks could reduce learning times below those of non-generalizing associative reinforcement learning algorithms.

The algorithm they describe is called complementary reinforcement backpropagation (CRBP). It operates by using a standard backpropagation network to map an input vector **i** to an output vector **s**. Since the units involved have sigmoid activation functions, the output vector **s** has continuous values in the range [0…1]. These are interpreted as probabilities that a corresponding bit in another binary vector **o** will have a value of 1. Vector **o** is then evaluated by a reinforcement function. Should this evaluation be positive then the mapping of vector **i** to vector **o** is positively reinforced by making vector **o** the target vector of conventional backpropagation. Vector **o** – vector **s** (the real valued current output) is used as the error vector. Should vector **o** receive a negative evaluation then the vector **s** is pushed towards the complement of vector **o** by taking ((1 – **o**) – **s**) as the error vector. This makes the mapping of vector **i** to vector **o** less likely.

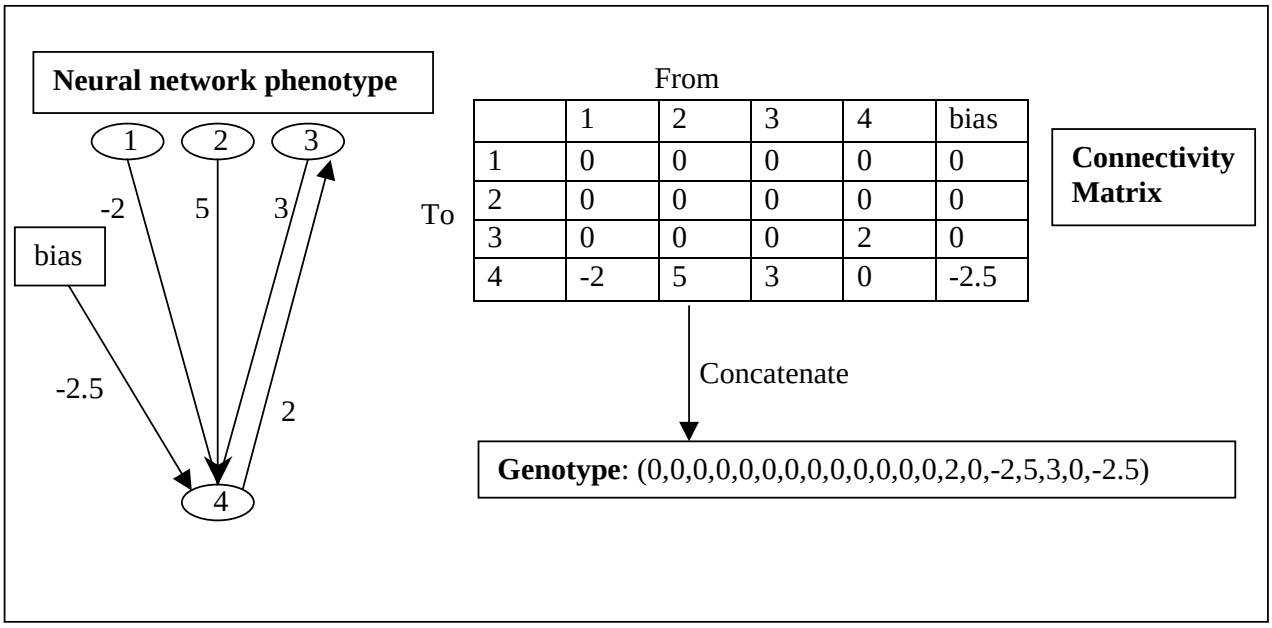
CRBP, then, combines a conventional supervised learning technique (backpropagation) to allow multi-layer neural networks to be trained via reinforcements. The following describes how Evolutionary Computation can be used to train neural networks.

2.3.3 Evolutionary Design of Neural Architectures

In nature biological neural networks evolved as powerful mechanisms of intelligence. Evolutionary computation and artificial neural networks are also highly compatible with each other. Unlike high-level computer languages that are brittle in the sense that small changes may result in radical alterations in the functioning of the program, neural networks can assimilate small changes in their structure by producing small changes in their behaviour. This has the effect of smoothing the fitness landscape in comparison to that produced by the solution space of a high-level programming language, thereby providing gradients that are more informative to an evolutionary process.

Evolutionary computation can evolve both topologies and weight settings of neural networks. A common method involves specifying the connections and their strengths by a connectivity matrix. This is then concatenated into a string of values and used as a chromosomal description (genotype) of the neural network (phenotype). Refer to “Evolutionary Design of Neural Architectures” by Balakrishnan and Honavar for a preliminary taxonomy and a guide to literature on the topic [Balakrishnan and Honavar, 1995a].

An example neural network and its connectivity matrix is shown in the following diagram (Figure 2-4).



	1	2	3	4	bias
1	0	0	0	0	0
2	0	0	0	0	0
3	0	0	0	2	0
4	-2	5	3	0	-2.5

Figure 2-4 A neural network and its genotype representation.

A genetic algorithm would use the chromosomes as follows:

(1) An initial population of chromosomes is created.
(2) They are decoded into their phenotype (neural network) representations.
(3) These are evaluated by a fitness function.
(4) A new population is created - parent neural networks are selected with a probability in proportion to their fitness, after which the genetic operators of mutation, crossover, inversion…etc, utilize the parent's neural network genotypes to create offspring.
(5) The process repeats itself (from 2 above) until acceptable performance levels have been reached or a predefined number of generations have elapsed.

Both supervised and reinforcement learning can be achieved using evolutionary computation. In the case of supervised learning a fitness function is used that evaluates the neural networks according to their success in minimizing the error between their behaviour and the known, desired behaviour. With reinforcement learning the fitness function evaluates the neural networks according to their success in maximizing the positive reinforcements they receive throughout their lifetime.

However, a fitness function is an evaluation of a neural network as a whole and, thus, operates at a distance to particular aspects of the network (like biological genes, they proliferate due to their statistical association with the success of the neural networks they are contained in). Individual neural network learning algorithms, though, can be more informative and target particular aspects of the neural network, thereby potentially making quicker adjustments. This does not mean that evolution or population level learning is rendered redundant. It has unique global learning

characteristics that other learning methods lack. Our work investigates the effects of combining such global and local learning processes in the context of neural network agents existing in a virtual environment.

2.4 Evolution and Learning in Virtual Environments

Much useful research combining evolution and learning investigates the adaptability of agents existing in virtual environments that are designed to provide the need for complex, adaptive behaviour. These environments are often of an artificial life type (consisting of food, poison, obstacles, predators, …etc) and are utilized by both artificial life and adaptive agent research.

Conventional neural networks have been investigated in simple environments where they map inputs to appropriate outputs with little effect on the external environment. Biological neural networks, though, display more complex behaviour. They exist in environments in which their outputs have an effect on their surroundings and, therefore, on their inputs. They must have the ability to interact with and adapt to their environment by taking advantage of the regularities they are exposed to, which includes behaving in ways so as to manipulate their own input. Thus, such neural networks must learn what effect their outputs have on their inputs in relation to the external environment. This, then, is an active relationship with an external environment that can require complex, intelligent, adaptive behaviour. This is also true of adaptive agents existing in complex environments.

Nolfi and Parisi describe some essential differences between classical neural networks and artificial life neural networks in their paper “Neural Networks in an Artificial Life Perspective” [Nolfi and Parisi, 1997]. They point out that artificial life neural networks exist in an environment, they have a body and they evolve. The fact that they evolve is not as relevant as the first two points, since evolution has been applied to conventional neural networks. The fact that they exist in an environment with which they can interact is, as in the case of adaptive agents, an important and distinguishing feature.

Related work and our own research that follows this section incorporates neural networks that are grounded in and have an active relationship with an external environment.

Chapter 3: Related Work Dealing With Learning and Evolution

3.1 Introduction

In nature the evolution of learning gave to life the ability to adapt more readily to variations in its environments and take advantage of short-term regularities that would otherwise be missed by genetic knowledge encoding mechanisms that take advantage of only long-term regularities. Simulations combining evolution and learning investigate the interaction between the two processes.

It has been shown that evolution and learning can complement each other as a result of their combined global and local search characteristics. Evolution can effectively sample large areas of a solution space, due to its global nature, identifying potential areas that can then be more adequately searched using a local learning process.

Evolution can also be used to optimize learning rate parameters, like velocity and momentum, according to the problem being solved.

Learning and evolution, though, can interact in a more complex manner with learning guiding the direction taken by an evolution process. This is known as the Baldwin effect [Baldwin, 1896] and occurs as a result of learning altering the fitness landscape of genetic solutions. It implies that should there be a cost of learning then there can exist a selective pressure to genetically predispose individual solutions with the required knowledge, since the faster an individual can acquire beneficial knowledge the fitter it is likely to be. In addition such knowledge may be difficult for a pure evolution process to discover. Learning, then, can be instrumental in initially discovering such knowledge and, as a result of selective pressure to encode the knowledge, make it easier for evolution to follow.

Learning, though, can also have the effect of obscuring genetic differences and, thus, hindering the evolution process. This is known as the Hiding effect [Mayley, 1997]. Essentially this occurs as a result of different genotypes mapping to the same or similar phenotypes (as a result of learning) with, should the cost of learning be low, equivalent fitness scores being produced. Thus, the genotypes cannot be effectively discriminated according to their fitness scores and the evolution of effective solutions is hindered.

However, certain situations require learning and evolution to be combined in order to solve problems that either alone cannot. This can be the case with highly dynamic environments where changes occur at a rate that is too fast for evolution or learning alone to effectively keep track of. In this respect Darwinian evolution and learning has been shown to be superior to Lamarckian evolution, although, the latter does have its advantages in more stable environments.

Since both methods can be superior in solving certain problems, work has been performed investigating hybrid Darwinian/Lamarckian evolution models.

The following presents these aspects of learning and evolution, and the related work dealing with them, in more detail.

3.2 Combining Global and Local search

Evolutionary algorithms can be used to perform a global search of a solution space, providing areas of high potential that can be further refined by the application of a local learning algorithm, thereby combining the advantages of global and local search methods.

In the context of neural networks an evolutionary algorithm can be used to globally explore the space of initial weight values, potentially aiding a proceeding neural network-learning algorithm in its local search.

Such searches can be effective due to a useful combination of global and local search characteristics. Evolutionary Computation search techniques are by nature global in character. Individuals in the population of solutions can be widely dispersed over the solution space. The individuals should only become more alike as the algorithm converges on an acceptable solution. The disadvantage of such techniques is that the local nature of the search may not be that effective. Potential solutions may be passed over, due to the coarseness of the search.

Conventional learning algorithms are local in character. The immediate neighbourhood of the current solution is effectively searched. Of course the disadvantage of such “short-sightedness” is the danger of falling into local optimums, where further movement results in worse solutions, even though better solutions are available in the solution space. This afflicts popular connectionist gradient descent type search techniques.

Such problems can be overcome by using hybrids of global and local search techniques. By maintaining a population of solutions, widely dispersed over the solution space, potential areas can be identified that can then be more effectively searched using local techniques.

A classic paper, authored by Belew, McInerney and Schraudolph [Belew, McInerney and Schraudolph, 1990], presented their results using a hybrid technique to combine a genetic algorithm with connectionist learning.

Their work involved a series of experiments combining the basic genetic algorithm with two different gradient learning techniques (conjugate gradient and backpropagation) used for training neural networks.

The genetic algorithm was used to find initial weight values from which the learning algorithms could further optimize. The results showed that these hybrids could dramatically reduce training times and find more effective solutions than the techniques applied in isolation.

Such hybrids are successful due to the genetic algorithm providing good “seeds” from which the gradient learning algorithm then continues to search. The genetic algorithm effectively samples global information, with its population of solutions dispersed over the solution space, while the gradient techniques effectively search the local neighbourhood surrounding each individual solution.

Belew, McInerney and Schraudolph concluded that:
"… the GA's global sampling characteristics complement connectionist local search techniques well, leading to efficient and reliable hybrids."

3.3 Optimizing the Parameters of a Learning Algorithm

Many learning techniques have critical parameters, like learning rate and momentum, which greatly affect the success of the search. Unfortunately, there is often no tried and tested technique for knowing what the most effective parameters are, as they vary according to the problem being solved. The experimenter's experience with the problem domain may be involved in selecting such parameters, or they may be selected according to a process of empirical experimentation. Another approach, though, is to apply an evolutionary algorithm to the selection of such parameters, thereby automating an otherwise difficult, tedious task of "trial and error".

Experiments were also conducted by Belew, McInerney and Schraudolph [Belew, McInerney and Schraudolph, 1990] using a genetic algorithm to optimize parameters of the backpropagation algorithm (learning rate and momentum).

The results of the experiments verified the genetic algorithm's ability to optimize parameters as well as producing a surprising value for the learning rate, which differed considerably from the conventional values used for the problem being solved. This is an example of the ability of genetic algorithms and other evolutionary computation techniques to make discoveries in areas where conventional thought may fail.

Evolutionary computation can search areas that may have previously been overlooked, since it does not have to be anchored by predisposed beliefs or rigid rules regarding the solution space, although valid problem domain knowledge can and should be used to bound the solution space where necessary.

3.4 Learning Can Guide Evolution (The Baldwin Effect)

Should there be a cost to learning then learning can guide evolutionary development by providing a selective advantage to predisposed individuals. The more innately predisposed an individual is to acquiring the learnt knowledge, the more it can limit its cost of learning (i.e., time wasted learning the solution).
Therefore, a selective pressure exists to genetically assimilate the knowledge that is initially provided by learning. This is known as the Baldwin effect.

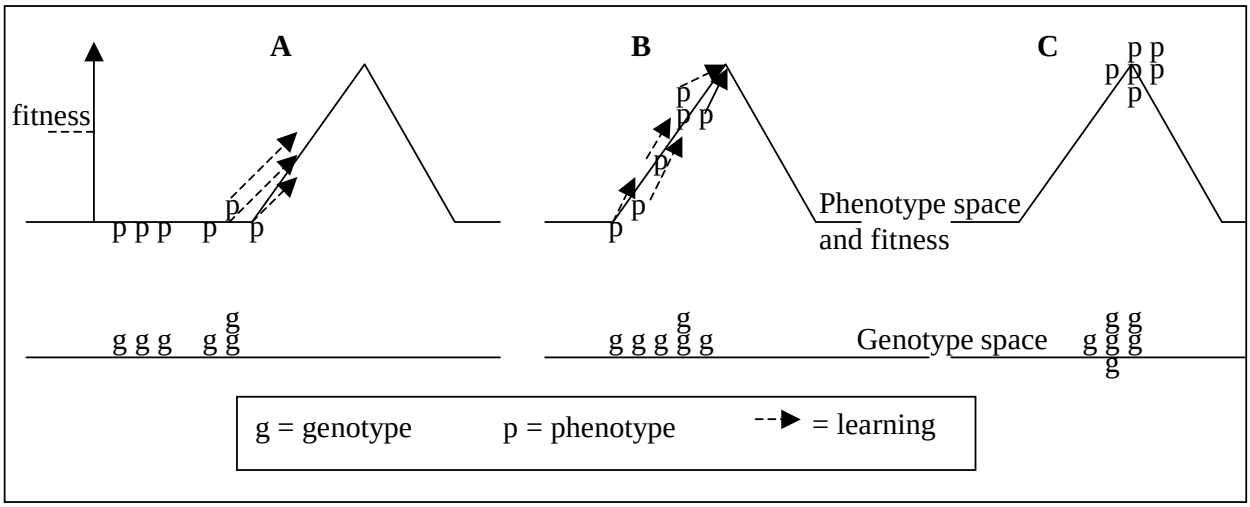


Figure 3-1 The Baldwin effect.

The above figure (Figure 3-1) demonstrates the movement of genotypes due to the Baldwin effect.

Initially only three genotypes translate into phenotypes that, due to learning, increase their fitness (A). They, therefore, have a selective advantage, which results in the following generation consisting of more genotypes inheriting similar positions and, thus, producing phenotypes being more predisposed to the peak being learnt (B). This continues until ultimately the optimal positions have been assimilated by the genotypes (C).

The Baldwin effect was first simulated by Hinton and Nowlan [Hinton and Nowlan, 1987].

In their model a genetic algorithm was used to evolve a population of neural networks with different inherited learning abilities. Each neural network consisted of 20 possible connections, with each connection being in one of three states: present, absent and learnable. These are represented by a 1, 0 and ? respectively. The task was to find the correct settings (a single binary string permutation) - clearly a “needle in a haystack problem” as there is only one correct setting in 2^{20} possibilities.
The learning method employed was a simple random guessing of 1 or 0 for each learnable connection.

Each network, then, was represented by a string of length 20 over the alphabet {0,1,?}. Individuals were given 1000 learning trials, with each trial consisting of random settings for the learnable connections. Fitness was measured inversely proportional to the number of trials needed to learn the correct settings. Individuals that could not find the correct settings had the lowest possible fitness. Individuals born with the correct settings would have the highest fitness. Their experiments using only evolution never discovered such an individual. However, with learning evolution was guided and successfully produced individuals with the correct settings from birth. The required knowledge was genetically encoded due to the guiding effect of learning.

Learning allows close solutions to be judged relative to the complete solution. With "needle in a haystack" problems like the one described, evolution cannot give partial feedback regarding how close a potential solution may be to the correct solution. Learning, though, has a smoothing effect on the fitness landscape allowing evolution to receive partial feedback and produce solutions it may otherwise find very difficult. This is illustrated by the following figure.

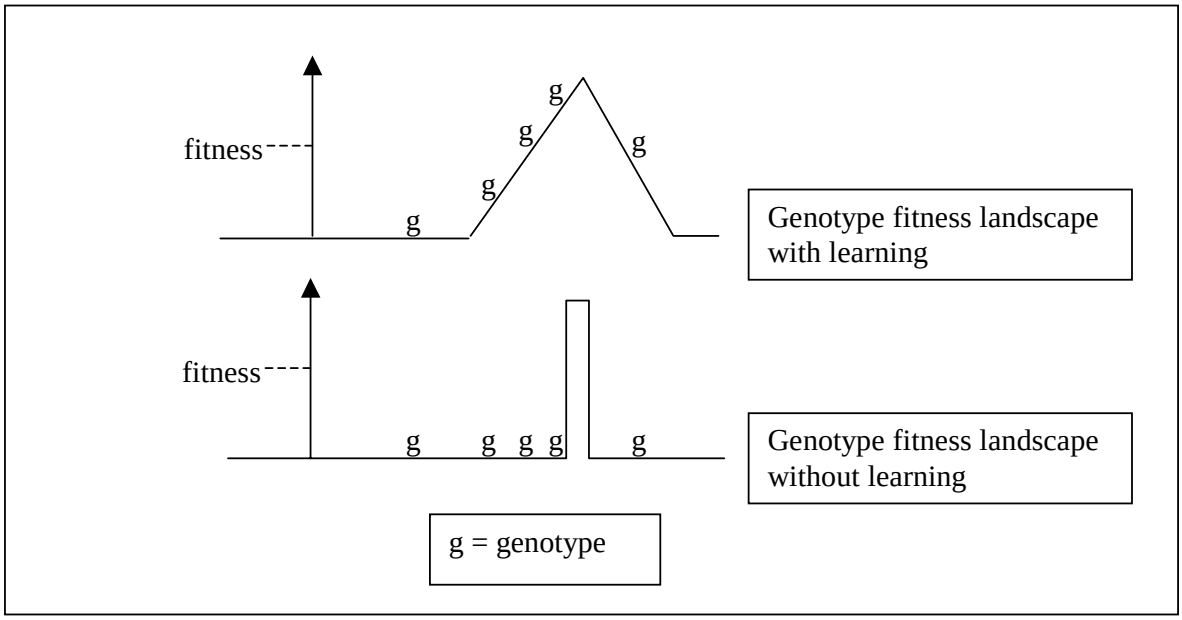


Figure 3-2 Learning smoothes the fitness landscape.

Figure 3-2 demonstrates how learning has a smoothing effect on the genotypes' fitness landscape by providing feedback regarding the position of genotypes that are near the desired "needle in a haystack" solution. Without learning such feedback is unavailable, making it difficult for an evolution process to find the correct solution. With learning, the genotypes can be guided towards the correct solution.

3.5 Learning can Hinder Evolution (The Hiding Effect)

In contrast to the Baldwin effect, learning can hinder the genetic assimilation of knowledge as a result of the Hiding effect [Mayley, 1997].

The Hiding effect occurs when learning hides genetic differences, reducing evolution's selective pressure. This reduction in evolution's ability to selectively discriminate between genotypes results in its ability to move the population in genotype space being compromised. This is demonstrated by the following figure.

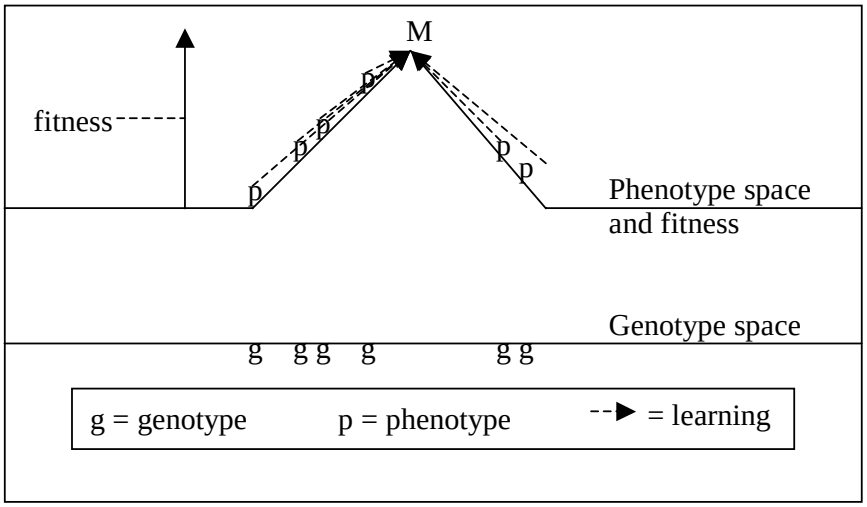


Figure 3-3 The hiding effect.

Figure 3-3 illustrates how one-dimensional genotypes (g) are translated into their equivalent phenotypes (p). Learning allows these phenotypes to move in phenotype space, reaching the same local fitness maximum (M). If fitness is only used as a selective measure at the end of a generation, regarding the final position of the phenotypes, then all phenotypes and their genotypes will receive the same fitness score. As a result there will be no selective pressure to move the genotypes towards producing innately fitter phenotypes. If learning did not occur, evolution would make use of the fitness differences of the innate phenotypes (p), rewarding phenotypes that are closer to the maximum and, thereby, resulting in a movement of genotypes towards this maximum. Learning, in this case, has hindered the movement of genotypes.

Since learning can both guide and hinder evolution there must be conditions that determine which effect is dominant. Mayley analysed the effect of two important variables: the cost of learning and epistasis.

His analysis showed that the degree of genetic hiding is inversely proportional to the cost of learning. This is easily explained. Should learning have high, detrimental fitness costs (i.e., the time wasted learning an effective solution), then there should be a selective pressure to genetically encode the required knowledge, producing innately fit individuals and reducing or eliminating learning times.

Should learning have low fitness costs then the ability of evolution to discriminate between genetic differences will be reduced, since different genotypes can produce the same phenotypes with little or zero fitness consequences. Thus, movement in genotype space will be compromised.

Mayley's analysis also showed that the degree of genetic hiding is also inversely proportional to the degree of epistasis involved.

Epistasis is a measure of the extent to which the fitness contribution of a particular gene is dependent on the effects of others. Low epistasis means that the effects of genes are relatively independent. High epistasis means that the effects of genes are highly inter-dependent. An implication of this is that a high degree of epistasis will result in a rugged fitness landscape, whereas a low degree of epistasis will produce a smoother fitness landscape (Figure 3-4).

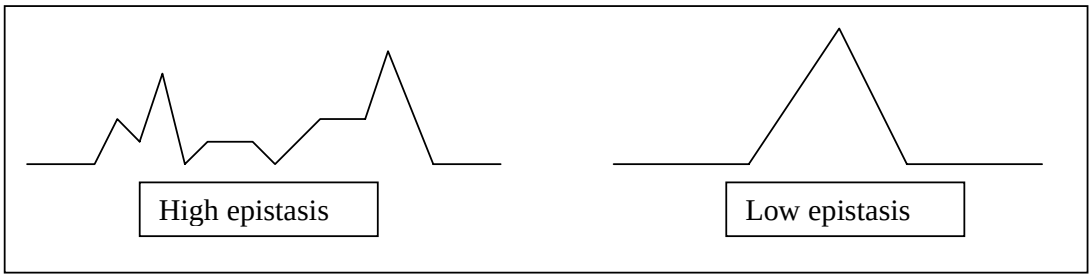


Figure 3-4 High and low epistasis.

A high degree of epistasis (rugged fitness landscape) is correlated with a greater dominance of the Baldwin effect, whereas a smoother landscape (low epistasis) can result in genetic differences being more easily hidden.

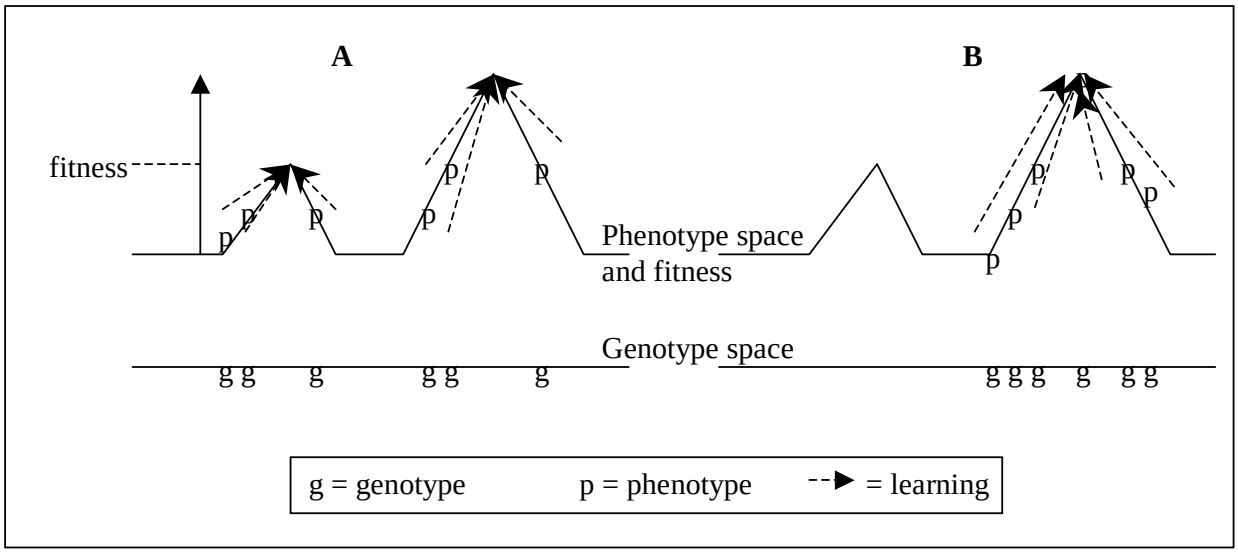


Figure 3-5 The hiding effect and epistasis.

Referring to Figure 3-5, learning will help the phenotypes reach their local peak (A). Those that learn the fitter peaks will give their genotypes a selective advantage, thereby moving the genotypes towards the fitter peaks, producing the Baldwin effect (B). Note that this movement can occur with evolution alone, but that learning may speed it up, since the fitter peaks may be more quickly identified.
Once the phenotypes are learning the same peak the Hiding effect will take over if the cost of learning is negligible (the genotypes cluster around the peak, but there is no pressure to move towards the optimal point).

Therefore, a rugged landscape will allow the Baldwin effect to dominate, utilizing learning to guide the movement of genotypes towards fitter peaks. Once these peaks have been identified, the Hiding effect may become more prominent if learning costs are low.

However, although learning may obscure the ability of an evolution search process to navigate solution spaces, evolution and learning can be combined to provide problem solving abilities that either method alone does not possess.

3.6 Evolution and Learning Can Solve More Problems Than Either Alone

Littman [Littman, 1995] used a simple environment model to illustrate how evolution and learning can interact to solve problems that either method alone cannot.

The environment is based on Todd and Miller's associative eater environment [Todd and Miller, 1991] and consists of individual agents who are born and live in a feeding patch. Their behaviour consists of choosing between substances that present themselves, which come in two forms (food and poison) and two colours (red and green). The association between colour and substance is dependent on the feeding patch while fitness is calculated as being the total amount of food consumed minus the total amount of poison consumed over a lifetime of L steps. The individuals must, therefore, maximize the amount of food they eat within their lifetime. The decision to eat or not to eat objects of a certain colour is determined by a simple neural network encoded into each individual's genes. Thus, optimal neural networks should map the colour of a food object to an eat response.

Evolution alone can find optimal behaviour if the substance to colour mapping is kept constant. For example, if food is always green and poison always red then neural networks will evolve to map green substances to an eating action and ignore red substances. Such a network is depicted below (Figure 3-6).

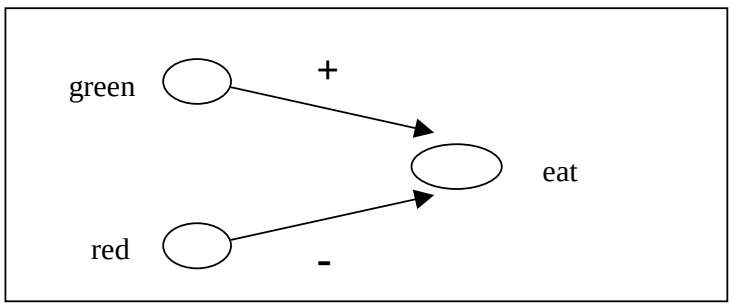


Figure 3-6 Neural network mapping of green/red objects to an eating response.

Green substances are shown to have a positive influence on the eating neuron while red substances have a negative (inhibitory) influence.

However, if individuals are born into one of two feeding patches uniformly at random with the substance colour relationship reversed (i.e., In patch A food is green and poison red with the reverse holding for patch B) then evolution alone cannot produce individuals who are a priori adapted, since these networks are fixed from birth. A learning mechanism would be required to find optimal behaviour within the individual's lifetime.

As previously discussed, combining learning and evolution can speed up the adaptation process. Littman alludes to this using the above model in the context of supervised and reinforcement learning. However, he also draws on Todd and Miller's work to demonstrate how combining the two processes can result in solutions with an effectiveness that neither alone can achieve.

Todd and Miller describe a network architecture that makes use of inherited knowledge and unsupervised (Hebbian) learning to achieve effective solutions where either process alone would not. In the context of the associative eater environment, consider a second input (smell) that distinguishes food (sweet smell) from poison (sour smell) in both patches, although with a probability of being misperceived (25% chance). An individual could be evolved to eat everything that smells sweet resulting in a 75% success ratio. Learning, though, could be used to associate the colour of the substance with its smell. For example, if food is green then eventually this input alone, via association, will be sufficient to elicit an eat response even though the food may be perceived to smell sour (25% of the time). The green input will override the sour smell, due to its association with the sweet smell and the eating response 75% of the time. Once the association has been learnt the individual will behave optimally (eat all green substances).

Evolution can extract the reasonably consistent smell relationship with unsupervised (associative) learning then utilizing this information to produce optimal behaviour. Either method alone would not result in an optimal adaptation.

Thus, evolution can extract useful knowledge regarding aspects of an environment that can be further utilized by learning to provide an effectiveness that either method alone cannot.

3.7 Comparing Lamarckian and Darwinian Evolution

Learning and evolution can be combined in two different frameworks, Darwinian and Lamarckian.

Lamarckian evolution is a theory of evolution that precedes Darwinism. It is the belief that knowledge learnt during an individual's lifetime can be inherited by its offspring. Although Lamarckian evolution has not been shown to occur in nature, with regard to changes in DNA, it can be simulated for evolutionary computation purposes.

The question is, does simulated Lamarckian evolution provide improvements over simulated Darwinian evolution?

Imada and Araki explored the Lamarckian evolution of associative memory [Imada and Araki, 1996] and found that their genetic algorithm utilizing Lamarckian evolution demonstrated improvements in the number of patterns that could be stored by a Hopfield neural network as well as the rate at which these patterns could be learnt.

Bryant A. Julstrom [Julstrom, 1999] compared the effectiveness of Darwinian, Baldwinian and Lamarckian strategies in solving the 4-cycle problem, which seeks the shortest collection of disjoint 4-cycles on a set of n = 4k points. A collection of 4-cycles is encoded as a single chromosome.

Learning was used to optimize single 4-cycle permutations within a chromosome and combined with a genetic algorithm to various degrees. Julstrom implemented 7 different strategies: Darwinian, Baldwinian one, Baldwinian some, Baldwinian all, Lamarckian one, Lamarckian some, Lamarckian all.

The Baldwinian strategies use learning to optimize 4-cycles in an individual chromosome, however, the optimized 4-cycles are not used to update the chromosome. Such chromosomes, though, are evaluated according to their improved fitness. The Lamarckian strategies do update the chromosomes with the improved 4-cycles.

The Darwinian strategy is a simple genetic algorithm without learning. Baldwinian one applies learning to a single, randomly chosen 4-cycle permutation. Baldwinian some applies learning to n/25 randomly chosen 4-cycles. Baldwinian all applies learning to all the 4-cycles in a chromosome. The Lamarckian strategies are applied in the same manner, except changes to 4-cycles are updated in the chromosomes.

These seven search strategies were applied to six different instances of the 4-cycle problem. Julstrom concluded that the most aggressive Lamarckian strategy, optimising all 4-cycles in a chromosome, consistently produced the best results. The Darwinian and the weaker Lamarckian strategies produced generally equivalent performances. The Baldwinian strategies provided the poorest results, their effectiveness deteriorating with an increasing use of learning. Julstrom explains this as being due to learning misdirecting the genetic algorithm's search (i.e., Hiding Effect). Thus, in this investigation the Baldwin effect was found to be completely

ineffective while Lamarckian evolution utilizing learning to the greatest degree produced the most effective results.

It seems natural to assume that population search methods that maintain learnt knowledge would demonstrate advantages over those that do not. However, this is not necessarily the case.

Whitley, Gordon and Mathias [Whitley, Gordon and Mathias, 1994] compared Lamarckian evolution with a Darwinian strategy using conventional genetic algorithm and learning without inheritance.

It was shown that the latter method, exploiting the Baldwin effect, can sometimes converge to the global optimum while Lamarckian evolution will converge to a local optimum. However, the Darwinian strategy was shown to be much slower for all their test cases. They also concluded that there are equivalence classes of functions that will be processed in the same way using either strategy. Thus, these results indicate that the effectiveness of either strategy depends on the nature of the problem being investigated.

Sasaki and Tokoro [Sasaki and Tokoro, 1997] investigated Darwinian and Lamarckian methods in stable and dynamic environments with regard to their ability to produce appropriate adaptive behaviour that could effectively follow the changes in an environment.

Their simulations made use of agents consisting of feed-forward neural networks that map binary patterns of 5 or 6 bits (representing two substances, food and poison) to output actions (eat or discard). They receive units of “energy” for eating food and lose an equivalent amount should they consume poison. Reinforcement learning is used to positively reinforce behaviour that results in their energy levels being increased while negatively reinforcing behaviour that results in decrements to their energy levels. At the end of each generation the agents are selected as parents with probabilities proportional to their energy levels. Thus, their final energy level is considered to be their fitness and is used by a genetic algorithm to create the following generation. Under their Darwinian strategy knowledge learnt during an agent’s lifetime is lost, while the Lamarckian strategy encodes learnt knowledge into the neural networks of offspring agents.

Thus, agents maximize their fitness by learning to discriminate between food and poison patterns via the processes of evolution and reinforcement learning, which are combined in both Darwinian and Lamarckian frameworks.

Sasaki and Tokoro confirmed that their Lamarckian strategy was far more effective than a Darwinian one in a static environment. In order to investigate dynamic environments they performed two experiments involving dynamic situations that change to a low and high degree.

The first experiment involved a situation where unknown patterns for food and poison appear and disappear over the course of generations. However, each pattern does not change between food and poison substances. That is, a pattern is consistently associated with a substance of a particular type.

The patterns are selected from a set of four patterns for both food and poison. However, only two of each are presented to the agents within any particular number of generations. Thus, individual agents only have a partial view of the environment (complete set of patterns).

Their results showed that the Darwinian agents were far more effective in correctly discriminating between the complete set of food and poison patterns by effectively integrating the partial information received over the generations. The Lamarckian agents' nature of adapting too greedily to a particular situation by passing on their adaptations to their offspring hindered their ability to learn the complete set of patterns. Thus, the Darwinian agents demonstrated a greater ability to adapt to the general characteristics of the environment as compared to the Lamarckian agents' nature of adapting too readily to specific situations, thereby hindering their ability to acquire general knowledge regarding the environment.

Their second experiment utilized a more dynamic environment. Here patterns representing food and poison were reversed at intervals of 50 generations. That is, food becomes poison and vice-versa.

The Lamarckian agents could not effectively adapt to such a situation, since they adapt too aggressively to eating food that ultimately becomes poison, resulting in unstable, oscillating fitness levels. The Darwinian agents, though, demonstrate an ability to adapt to such situations, producing stable fitness levels that improved over the generations.

Sasaki and Tokoro also noted that the innate abilities of the Darwinian agents, prior to learning, seemed to decrease as the generations progressed, indicating that learning ability is being favoured over innate, performance ability. This led them to conclude that the ability to perform something is not as important as the ability to learn in dynamic environments. Lamarckian evolution emphasizes the former while Darwinian evolution seems more suited to producing the latter. As a result, Darwinian evolution demonstrates a greater adaptability towards dynamic environments.

Thus, to conclude this section, it seems that Darwinian and Lamarckian strategies have different adaptive characteristics suited to different types of problems. This provides the opportunity of creating hybrids that effectively combine the advantages of both methods.

3.8 Hybrid Darwinian/Lamarckian Models

Sasaki and Tokoro extended their research of the previous section by introducing an heredity rate parameter into their evolution model [Sasaki and Tokoro, 1999]. This parameter controls the degree to which agents inherit the knowledge of their parents. As a result, the amount of Lamarckianism or Darwinism occurring in the population can be specified. Since their previous work indicated that Lamarckian methods could adapt quite quickly to particular situations while Darwinian methods indicate a greater adaptability towards dynamic environments, the heredity rate was investigated in terms of its ability to provide partial Lamarckian/Darwinian mechanisms to cope with dynamic environments while still quickly adapting towards particular situations.

The heredity rate, r, was utilized as follows:

$W_c = W_o + r.(W_l – W_o)$

where W_c is a new offspring chromosome's vector of connective weights, and W_o and W_l represent the parent's vectors of connective weights at the time of birth and at the time of reproduction (after learning).

Thus, with r = 0 individuals do not encode any learnt knowledge into their chromosomes, i.e., fully Darwinian. With r > 0 knowledge is inherited to a certain degree according to r. When r =1 the individuals are fully Lamarckian; they inherit all learnt knowledge. Individuals with $0 < r < 1$ are, therefore, partially Darwinian and partially Lamarckian.

Sasaki and Tokoro confirmed that when applied to static environments the populations with higher heredity rates produced the best performances, with a full Lamarckian strategy being the most effective.

When applied to dynamic environments, of the type described in their previous work, the opposite was true. Increasing levels of Darwinism were found to be the most effective in adapting to the changes in the environment.

Thus, the results of their previous paper were validated: Lamarckian methods are highly effective in adapting to static environments, while Darwinian methods are more effective when dealing with dynamic environments.

It remains to be seen whether the use of a hybrid Darwinian/Lamarckian strategy can result in improvements over full Darwinian or Lamarckian methods.

Houck, Joines, Kay and Wilson have also investigated the benefits of partial Lamarckianism [Houck, Joines, Kay and Wilson, 1997].

Their model utilizes a heterogeneous population of Lamarckian and Darwinian individuals. For example, 50% of the population can be Lamarckian, with their genotypes being updated to incorporate learnt knowledge.
This strategy was applied to seven different test problems using different percentages of Lamarckian and Darwinian individuals, from 0% Lamarckianism (i.e., a pure

Darwinian strategy) to 100% Lamarckism. In addition, a pure Darwinian strategy without learning was also evaluated.

Their investigation concluded that neither the pure Darwinian nor pure Lamarckian strategy was found to be consistently effective. It was discovered that the 20% to 40% partial Lamarckian strategy was the most effective in minimizing worst case performances across the entire set of test problems.

Thus, these results suggest that using both Darwinian and Lamarckian individuals in a population can lead to improvements in searching efficiency.

This concludes our related work section. The following chapter describes the environment in which our evolution and learning models are investigated.

Section 2 - Our Work

Chapter 4: Environment, Agent Architecture and Method

4.1 Introduction

The environment and agents were designed to provide an effective means of evaluating our evolution and reinforcement learning models.

In order to achieve this, several criteria needed to be satisfied:

- The environment must require intelligent, adaptive behaviour that can be used to effectively judge the performance of our evolution and reinforcement learning models.
- It should be possible to evaluate our evolution and reinforcement learning models with regard to changes in environmental stability.
- Agents require an architecture suitable for both evolution and learning in Darwinian and Lamarckian frameworks.
- Agents should be designed so as to provide information regarding the nature of their adaptive knowledge, both innate and learnt.

The following describes our environment and agent architecture designed to satisfy the above criteria, concluded by a description of our method of investigation.

4.2 World Model

The environment consists of a population of agents interacting with food and poison objects. Agents are each given their own territory (Figure 4-1) in which they are allowed to move. They navigate their region by choosing from a set of three actions: move forward, rotate left and rotate right, each action being chosen in response to input from a 360° visual system, providing information concerning the angle and distance of objects in their territory.

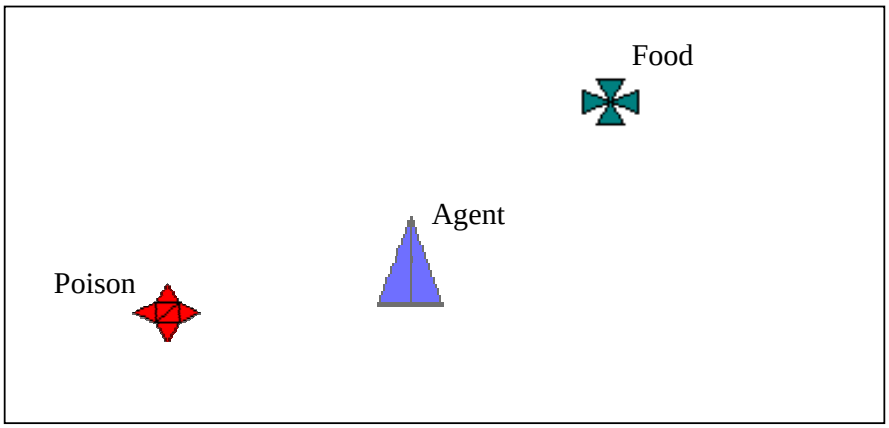


Figure 4-1 An agent's territory.

Initially, equal amounts of food and poison are placed in each agent's territory. On encountering food or poison they automatically eat it. The food or poison object is then immediately replaced at a new random position that is sufficiently far away from the agent to ensure that no free-meals occur. Thus, a specified number of food and poison objects are always present in an agent's territory. It is the objective of the evolution and learning models to produce agents that maximize their success in obtaining food objects while avoiding the poison objects.

In the case of our evolution model, eating food increases an agent's fitness and poison decreases it. Our reinforcement learning model provides positive reinforcement for eating food and negative reinforcement for eating poison. Our Darwinian and Lamarckian models utilize both forms of feedback.

Agents are placed into the world with no initial knowledge as to what appropriate behaviour is. They, therefore, have to evolve or learn all the knowledge required to produce effective behaviour. A trace of an agent successfully making its way towards a food object is shown in the following figure (Figure 4-2).

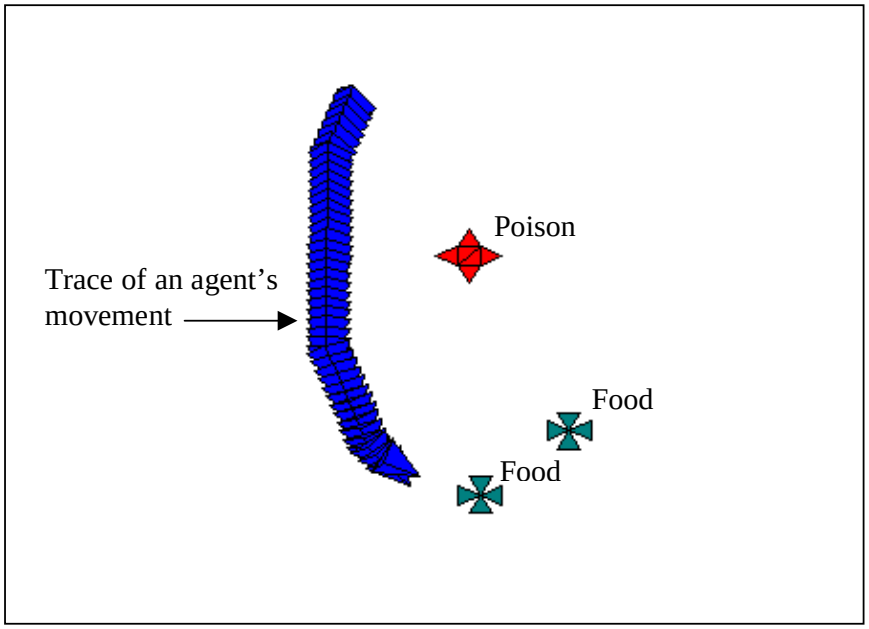


Figure 4-2 Trace of an agent successfully locating a food object.

Changes in environmental stability are achieved by alternating, at specified intervals, the relationship between an object and its fitness or reinforcement consequences (i.e., food becomes poison and vice-versa). Thus, a particular degree of stability can be specified according to a chosen frequency of fitness alternations. This allows us to evaluate and compare the ability of our models to adapt to environments of changing stability.

4.3 Agent Architecture

Agents need to perceive their environment, process what they "see" and act on it. This requires a visual system, a processing system and a repertoire of actions (Figure 4-3).

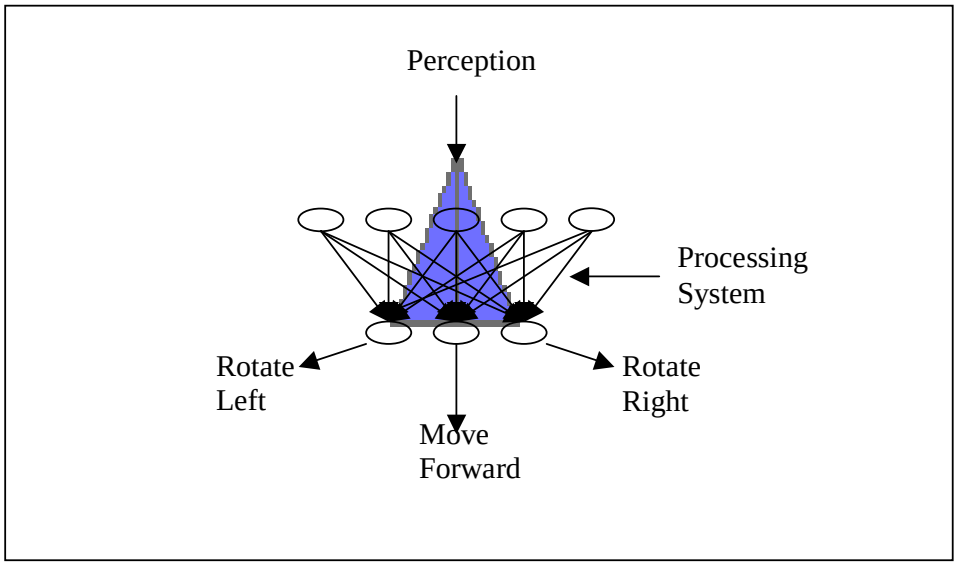


Figure 4-3 The agent's architecture.

4.3.1 Visual System

An agent's vision has a 360° range. The distance and angle to an object is calculated by a given function. Closer objects of the same type obscure those that are behind them; however objects of a different type do not obscure each other (i.e., food is visible even if it is behind poison). This is possible as the neural network processing system deals separately with each object type, by allocating a certain number of the agent's neural network input neurons to each object type. Thus, objects of different types do not obscure each other as they use a different part of the processing system's input. This gives the agent the most amount of information possible, given the constraints of the architecture.

It is the responsibility of the visual system to calculate the distance and angle of objects, sort objects of the same type according to their distance (removing objects that are obscured) and provide the information in an appropriate format for the neural network processing system.

4.3.2 Processing System

The processing system is a single-layer, fully connected, feed-forward neural network. This provides a suitable means of mapping an agent's visual input regarding the two object types (green and red objects) into an output action.

The information produced by the visual system is converted into neuron activations for the input layer, with the degree of activation of a particular neuron corresponding to the seen object's angle and distance. Each neuron is responsible for a portion of the agent's 360° vision and is activated according to the distance of the object. The closer the object, the greater the degree of activation.

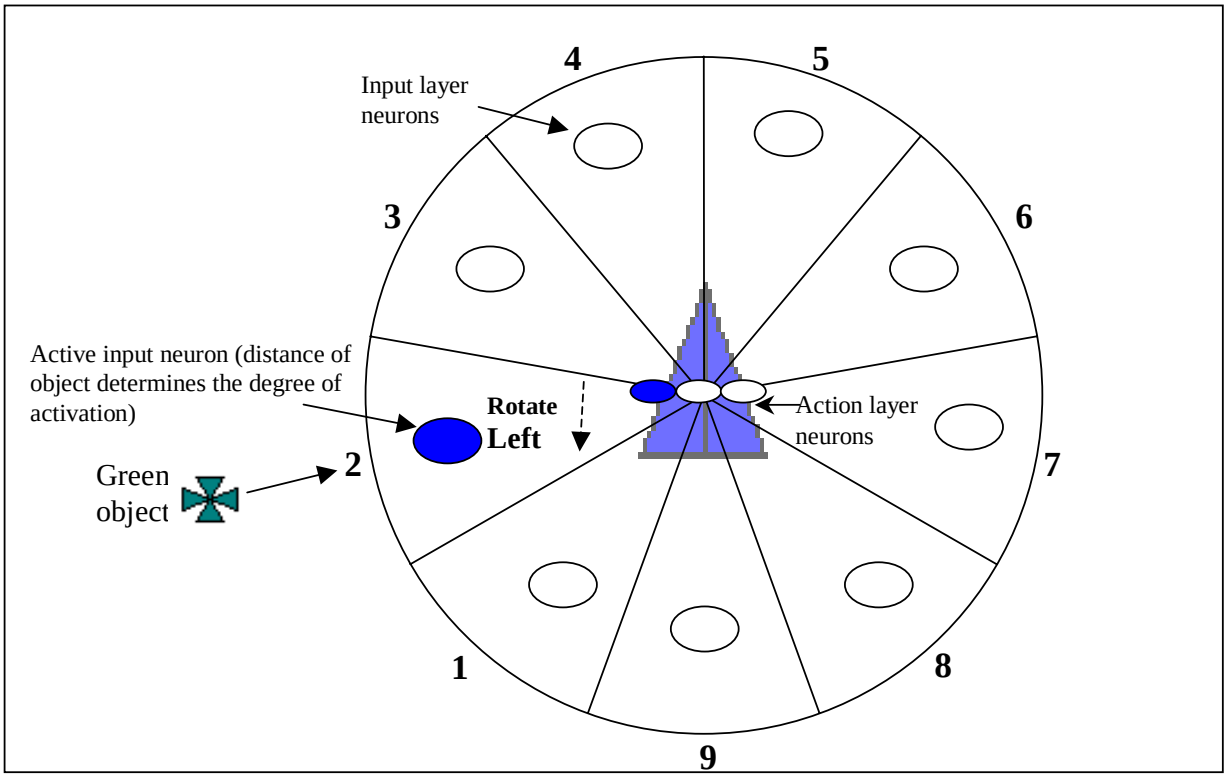


Figure 4-4 The position of food is mapped via the visual system to one of the agent's 9 input neurons. In this example a left rotation is chosen, thereby orientating the agent towards the food.

The above figure (Figure 4-4) represents an agent's visual input regarding green objects and its mapping to an action. A separate, but equivalent input neural network layer is used for red objects, with both layers mapping to the same action layer. Thus, a chosen action is the result of the positions of both green and red object types.

The number of input neurons determines the visual system's resolution (9 are used in the example above, giving 40° per neuron). Which neuron is activated depends on the angle of the object (i.e., neuron 5 covers the range [0°…40°], neuron 6 covers the

range [40°…80°]…etc). The appropriate neuron is then activated according to the distance of the object.

The figure below (Figure 4-5) corresponds to the above situation and represents an agent's neural network and its mapping of the position of the green object to the output neuron responsible for rotating the agent in an anti-clockwise direction, thereby orientating itself towards the object.

The neural network weights, that encode the agent's behaviour, are randomly initialised with values of 1, -1 and 0 in order to provide initial sampling points for our evolution model's search. Thus, the neural network below is an agent's initial network prior to evolution or learning.

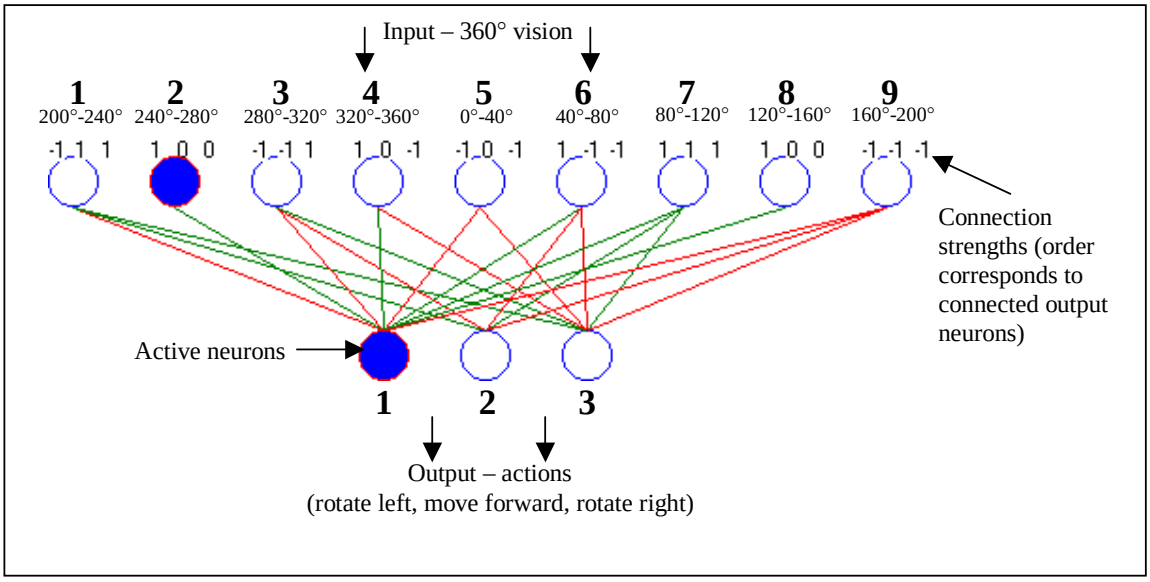


Figure 4-5 The agent's neural network portion responsible for mapping visual input regarding green objects to an output action.

Since the action layer is competitive, only one action can take place at a time. Inputs are summed according to:

$$X_j = \sum_{i}^{n} W_{ji} V_i$$

where X_j is the total sum of inputs received by neuron j (in the output layer),
W_{ji} is the weight of the connection from neuron i (in the input layer) to neuron j (in the output layer),
n is the number of neurons in the input layer,
and V_i is the activation level of neuron i (in the input layer).

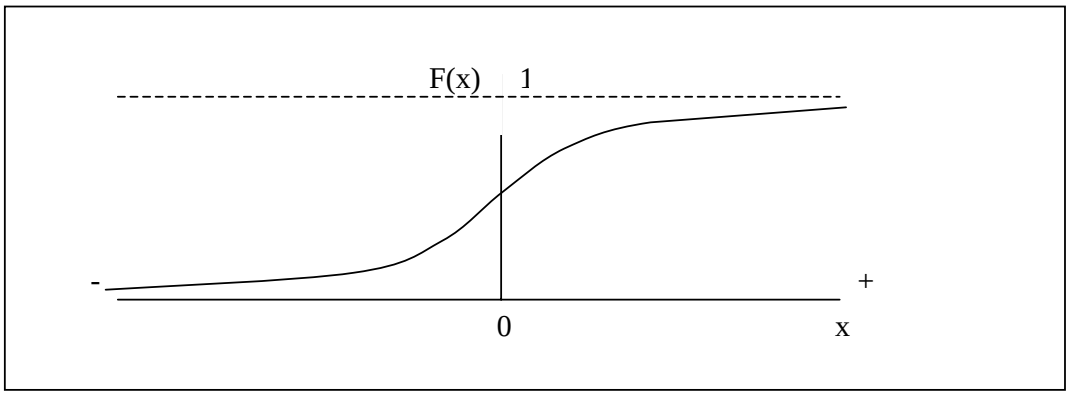


Figure 4-6 Binary sigmoid activation function.

Activation levels are determined by a binary sigmoid activation function (Figure 4-6):

$$O_j = 1/(1 + \exp(-\sigma X_j))$$

where O_j is the activation level of output neuron j,

σ is an optional steepness parameter for the slope of the sigmoid function,

X_j is the total sum of inputs received by neuron j.

The activation levels ($0 \leq O_j \leq 1$) are interpreted stochastically as probabilities that the output neuron will take on the value 1. When x equals zero there is an equal chance of the neuron being 0 or 1 (50%). If no neurons are chosen to be active or if more than one are active then the neuron with the greatest probability of being active is chosen. If the maximum probability of activation is shared then one of these is randomly chosen.

Probabilistic activation functions are used as they can help the networks to explore different mappings that would otherwise not occur using deterministic activation functions. The extra variation is useful since it adds an element of randomness that allows the network, every now and again, to try something else. This is related to the concepts of local and global optimums, as well as the issue of exploration versus exploitation found in the reinforcement literature. The randomness allows the network to move out of local optima (exploiting sub-optimal solutions) and explore other possibilities that may result in better solutions.

Richer possibilities for behaviour can also be encoded. A deterministic mapping will always result in the same response, given the same input. A probabilistic mapping allows the agents to respond to a particular input with actions that have different probabilities of occurring. For example, an agent's response to a particular orientation towards food may be to move forward 75% of the time and rotate to its right 25% of the time. A situation like this could occur with the food activating the agent's input neuron responsible for its 0° to 40° region. Since there is only one neuron (neuron 5) responsible for this region, the agent cannot refine its orientation to any greater level

of detail. It would be appropriate not always to move forward, but also to assign a certain probability of rotating to its right.

The connection weights are limited to maximum and minimum values so as to constrain the search space to an acceptable size. We use a maximum of 10 and a minimum of –10 for all our simulations. Thus, our evolution and reinforcement learning models search the same solution space.

In the case of our Darwinian agents, learnt changes to weights are lost, and inherited weight values are used to produce offspring. This provides a separation of learning at a genetic and individual learning level. Our Lamarckian agents inherit the learnt knowledge of their parents.

The weight settings are used to provide information regarding the adaptive nature of the genetic and learnt knowledge produced by our evolution and reinforcement learning methods. Since we are using single-layer, feed-forward neural networks, this information can be easily extracted.

This, then, is the basic neural network setup that each agent is given. More information regarding extensions to the architecture, necessary for our evolution and reinforcement learning models, will be given in the appropriate sections.

4.3.3 Action System

Each agent has three output neurons that map to one of three actions: move forward, rotate right and rotate left. The agents construct their behavioural routines using these fundamental actions as primitives.

Eating automatically occurs when the agent comes into contact with food or poison objects.

4.4 Simulation Constants

The following simulation constants are used for all experiments:

- Population size = 25
- Agent lifetime = 500 cycles
- Duration = 200 generations per trial
- Trials = 30
- Number of food and poison objects in the agent’s territory at any time = 1

4.5 Method

The three important objectives of our research are:

- To design several evolution and reinforcement learning models with a view towards maximizing their effectiveness in an initially stable environment that becomes increasingly unstable.
- To evaluate the adaptive nature of such models in response to environments of changing stability.
- To compare the performance of our models in response to various degrees of environmental stability.

The following describes our approach towards achieving these objectives.

4.5.1 Design

Our design objective is to maximize the effectiveness of our evolution and reinforcement learning models in stable environments, but with a view towards also maintaining adaptability to changes in such environments. To achieve this objective we utilize a stable environment to provide feedback regarding important design considerations. That is, mechanisms and parameters are evaluated with regard to their performance in the stable environment. We accept, though, that different mechanisms and parameters may be more suited to particular levels of stability. Thus, mechanisms are considered that can automatically adjust important parameters according to the environment's degree of stability, thereby providing a potentially greater adaptability towards changes in environmental stability.

4.5.2 Evaluation

The performance and adaptive nature of our models is evaluated in environments that range from stable to unstable.

The stability of our environments is determined by the frequency of food/poison fitness alternations. These range from completely stable (no alternations) to highly unstable (every generation).

Three measures of performance are used: two measures with regard to the maximum average fitness levels produced by the agents and a third measuring the sum (area) of the average fitness levels over the course of the simulation.

In addition to these measures, the knowledge produced by our models (neural network weights) is used to provide insights into their adaptive nature.

4.5.3 Comparison

Finally, we compare the performances and adaptive nature of our evolution and reinforcement learning models.

Chapter 5: Design and Evaluation of an Evolution Model

5.1 Introduction

This chapter will present the design and evaluation of our evolution model.

Our evolution model is designed to produce effective, adaptive behaviour in agents whose objective it is to consume food and avoid poison in a virtual environment. It is evaluated in terms of its performance in the environment demonstrating stable characteristics and in terms of its response to increasing levels of instability.

The environment poses an adaptive problem that allows us to demonstrate how our simulated evolution model can produce complex, adaptive behaviour utilizing limited feedback as to what the appropriate behaviour is. The feedback in this situation is an increase in an agent's fitness if food is consumed and a decrease in the case of poison. This is the only feedback provided to create agents that can navigate towards food and avoid poison. It is up to our evolution model to utilize the fitness scores in order to evolve effective agents.

Equivalent feedback is provided for our reinforcement learning model and our combined evolution and reinforcement learning models.

In our example problem behaviour is realised as a result of the knowledge that is encoded into each agent's neural network. The encoded knowledge provides mappings from input states (the agent's perception of the environment) to output actions (our agents can choose from moving forward, rotating left or rotating right). The agents are autonomous, that is, their behaviour is a sole result of their neural network mappings and they have no a priori knowledge regarding how to survive in their environment. The knowledge that will produce effective behaviour must be completely evolved.

In order to evaluate the performance of our evolution model, its effectiveness is observed in response to increasing levels of environmental instability. This is achieved by alternating the roles of food and poison objects with an increasing frequency according to the required degree of instability. Thus, the adaptability of the model can be effectively evaluated with regard to increasing levels of instability.

Our objectives, then, are to:

- Design an evolution model with a view towards maximizing its ability to produce adaptive agents in an initially stable environment that becomes increasingly unstable.
- Evaluate the effectiveness and adaptive nature of the model with regard to its performance in the stable environment and in terms of its response to increasing levels of instability.

5.2 Evolution

In nature the ultimate, implicit goal is survival. All other instrumental or approximate goals are derived from this ultimate goal. In simulating evolution we can attach our desired goals to the survival probabilities of the units being operated on. In this way their implicit goal of survival (imposed upon them as a result of selection) and our desired goals are correlated. The degree to which they achieve our goals is proportional to the degree to which they survive. Our goals become their instrumental goals to be achieved as a means towards maximizing their fitness and, thus, their survival.

Simulating evolution requires modelling certain essential qualities of natural evolution. The two fundamental components are selection and variation. Selection for reproduction has the effect of amplifying good solutions while culling bad ones. Variation has the ability to create new solutions by altering old ones, thereby searching new areas in the solution space. Variation can be achieved as a result of processes like crossover or mutation. Crossover is the process of combining aspects of solutions to create new ones, whereas mutation is the process of changing a previous parent solution by a small, random degree to produce new offspring solutions.

Simulated evolution algorithms utilizing selection and variation generally proceed as follows:

```
//initialise a random population of individuals
t=0;
Initialise P (t);

Repeat
  //evaluate the fitness of the individuals in the population
  Evaluate P (t);
  //select parents based on their fitness scores
  Parents = Select Parents from P (t);
  //create a new population by reproducing the parents using crossover and mutation to
  //provide variation
  P (t+1)′ = crossover Parents
  P (t+1) = mutate P (t+1)′
  //increment the generation counter
  t=t+1;
Until termination criteria have been satisfied
```

Note that populations P(0) to P(t-1) are discarded, since only P(t) and P(t+1) are utilized at any particular time.

5.2.1 Selection

Simulating selection requires that our agents have a fitness assigned to them. This is a measure of their ability to achieve our goals.
To achieve this evaluation a fitness function needs to be created.

5.2.1.1 The Fitness Function

The fitness function needs to measure the performance of a solution (our agents) relative to the goals that are to be achieved.

In the example problem our goal is to create agents that maximize their fitness by consuming as many food objects as possible while avoiding poison objects.
This is accomplished by assigning a positive fitness consequence to food objects and a negative fitness consequence to poison objects. The resulting fitness of the agent is then determined using a simple fitness function that sums the amount of food eaten less the amount of poison. Thus, our goal for the agents to eat food and avoid poison has been directly correlated with their survival chances.

This is the fitness function:

Fitness = x.f + y.p

where x is the amount of food eaten, f is the fitness consequence for eating food, y is the amount of poison eaten, p is the fitness consequence for eating poison.

Note the simplicity of the fitness function, describing our goal, while considering the potentially complex behaviour that may be required in the virtual environment to achieve this goal. A more complex fitness function could be used to provide more feedback regarding the agent's behaviour, however, the above fitness function demonstrates that a goal can be simply described even though it may require complex behaviour to achieve it.

Also, note that some aspects of a desired solution, such as reaching the food in the shortest amount of time, will be implicitly encoded in the fitness function, since minimizing the time taken will maximize the amount of food that can be eaten. There is no need to explicitly use time taken or distance travelled as parameters for the fitness function. Both of these will be naturally minimized, due to competition.

The agents accrue fitness over a fixed time period, one generation. An agent's fitness is only used after a complete generation has passed. It is then up to a selection mechanism to use the fitness scores to produce a new population.

5.2.1.2 Probabilistic Selection

The generation of a new population is influenced by Holland's suggestion that parents should produce offspring according to probabilities that are in proportion to their relative fitness [Holland, 1975]. This insight has been reinforced by Galar [Galar, 1985] who suggested that emphasizing less-fit individuals in an evolutionary simulation can be beneficial in escaping evolutionary traps. A probabilistic selection

mechanism gives less-fit individuals, who may be searching other promising areas of the solution space, an opportunity of continuing their search, rather than prematurely concentrating the search on a particular area.

5.2.1.2.1 Roulette Wheel Selection

Probabilistic selection can be accomplished using roulette wheel selection [Holland, 1975] – space is allocated on a "roulette wheel" in proportion to the agent's fitness.

The problem with roulette wheel selection is that it depends on positive fitness values. We solve this problem by first shifting all the fitness values into a positive range, keeping their relative relationship, and then adding up their values to give a total. Their share of the total determines their probability of reproduction.

Each agent's fitness is adjusted as follows:

$$F_A = F - F_M$$

where F_A is the agent's adjusted fitness, F is the agent's original fitness, and F_M is the fitness of the worst agent in a generation.

Their probability of reproduction then becomes:

$$R_P = F_A / \sum_{i=}^{n} F_{A\,i}$$

where R_P is the agent's probability of reproduction, F_A is the agent's adjusted fitness, and n is the size of the population.

For example, four agents with the fitness values: -5, -2, 4, and 8 would have them shifted to become: 0, 3, 9 and 13. This gives a total of 25 and, therefore, probabilities of: 0%, 12%, 36% and 52%.

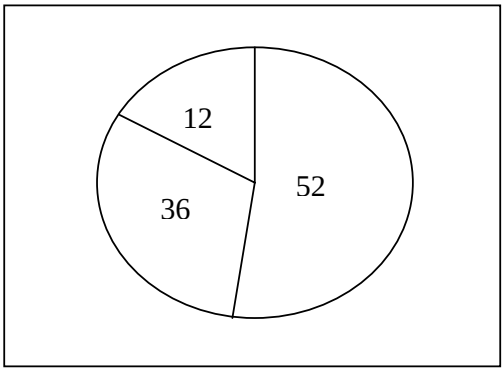


Figure 5-1 Roulette wheel selection – three individuals have been allocated space on the wheel in proportion to their fitness.

As each agent is evaluated its chance of reproducing is determined by the space given to it on the roulette wheel (Figure 5-1). In our model, reproduction is a process of copying and then possibly mutating the original parent to produce a new offspring individual.

All individuals are evaluated and the process repeated until a new population has been produced.

5.2.1.2.2 Barometer Selection

We created a second selection mechanism to place a greater emphasis on fitter individuals. The agent's probability of reproduction was altered to become:

$$R_P = F_A / F_R$$

where R_P is the agent's probability of reproduction, F_A is the agent's adjusted fitness, and F_R is the range in which fitness values exist.

The figure below (Figure 5-2) represents the strategy for the fitness values of: -5, -2, 4 and 8.

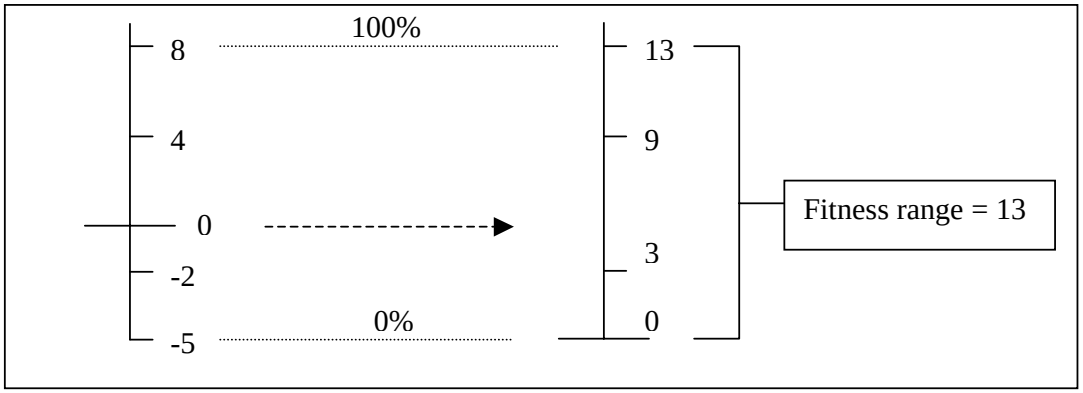


Figure 5-2 Barometer selection – Individuals are assigned probabilities of reproduction according to their position within the fitness range, given by the minimum and maximum fitness values found in the population.

An agent's probability of reproduction is then determined by its position on the fitness barometer. Individuals reaching the maximum fitness are guaranteed a place in the following generation, since their probability of reproduction equals 1. Individuals with the minimum fitness cannot contribute to the new population, since their probability of reproduction equals 0.

Using the described example, the individual with the highest fitness of 8 will be guaranteed to reproduce each time it is evaluated (100% chance of reproduction). On the other hand, the roulette wheel selection method will give the fittest individual the

highest probability (52%) of producing offspring each time it is evaluated for reproduction; however, this does not guarantee that it will produce offspring. In addition, since the individuals may be evaluated several times for reproduction the effect will be amplified. The fittest individual under a barometer selection method will always produce offspring each time it is evaluated, since the old population is repeatedly evaluated until a new population has been produced.

Thus, using the barometer style selection method there is a greater emphasis on reproducing fitter individuals while still giving less fit individuals the opportunity of reproducing.

5.2.1.3 Elitist Selection

A problem with probabilistic selection is that the fittest individuals could be lost as a result of not being selected for reproduction or due to the genetic operators (mutation, crossover…etc) changing the previous solution with detrimental results. The barometer style selection method is only vulnerable to the second case, since the fittest individuals are always chosen for reproduction.

In order to ensure that the fittest individual in a generation survives without changes into the next generation, we employ an elitist type strategy [Grefenstette, 1986] together with our barometer and roulette selection methods. This guarantees that the fittest individual will be reproduced (without changes) into the following generation. However, its fitness will be reset and it must compete for survival like any other agent (i.e., it is only as good as its last performance). This is to ensure that an individual does not gain a lasting hold on the elitist position due to favourable environmental circumstances, as compared to skilled behaviour.

5.2.2 Simulating Variation

In order to provide the necessary variation required of our evolution model we utilize a mutation operator. This allows new solutions to be created from old ones by mutating (randomly altering) aspects of their definition. Thus, it is a random process. However, when combined with selection it has the effect of moving the solutions towards areas of greater fitness.

5.2.2.1 Mutation

Mutation simply involves changing a solution by a random amount proportional to a mutation rate, normally small. For example, a binary string 111111 may produce the offspring binary string 111101.

Mutating the population, then, involves selecting individuals and then altering, by small random amounts, aspects of their definition. This has the effect of moving those individuals in the solution space. The new individual will then provide feedback regarding its position in the solution space, in the form of its fitness. These feedback/fitness values of the population's individuals will be used by selection to create the next population and the process repeated. Thus, selection of the fittest and variation through mutation drives our population's search.

5.2.2.1.1 Procedure

The probability of an individual agent being chosen for mutation and the degree to which it is mutated depend on two parameters: a population mutation rate and an individual mutation rate. The population mutation rate is a parameter that determines the degree of mutation that occurs in the population. The individual mutation rate determines the degree to which an agent will be mutated if selected for mutation. It can be a parameter to the system or it can be included in the agent's description (see section 5.2.2.1.3.3 dealing with self-optimization). Thus, selection for mutation occurs at two levels. Firstly, an agent is selected for mutation with a probability determined by the population mutation rate and then, secondly, aspects of the agent's definition are selected for mutation with a probability determined by the individual mutation rate.

Using our evolution model two attributes of an agent are available for mutation, its neural network and in the case of self-optimization, its own mutation rate.

5.2.2.1.1.1 Mutating an Agent's Neural Network

The knowledge that produces our agent's behaviour is encoded in its neural network weights or connection strengths between the modelled neurons. Thus, in order to alter our agent's behaviour we need to mutate its neural network connections.

Connections are chosen for mutation with a probability determined by the individual mutation rate. Should a connection be chosen it is mutated by altering its weight by a small random value.

The following diagram (Figure 5-3) demonstrates the mutation of two connections.

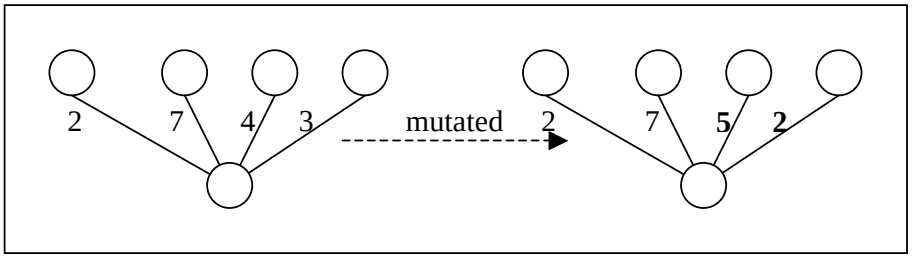


Figure 5-3 A simple neural network is mutated by randomly selecting and changing two of its connection weights.

The above neural network in Figure 5-3 has been mutated by having its third and fourth connections mutated by 1 and –1, respectively.

5.2.2.1.2 Mutate Once vs. Repeated Mutations

We have investigated two methods of selecting individuals and their neural network connections for mutation.

- Individuals and their neural network connections can be selected more than once for mutation. That is, after each mutation they can be selected for further mutations until the new population has been mutated to the degree determined by the mutation rates and is ready for evaluation.
- Individuals and their neural network connections are made unavailable for selection once they have been mutated. Further mutations must be applied to neural network connections and individuals who have not previously been mutated.

The possibility of repeated mutations could have the effect of mutating an individual within a larger range, since it may be mutated more than once, whereas the degree of mutation experienced by an individual that can only be mutated once will be more consistent. Its range of mutation will be more directly related to the individual mutation rate. This results in the required degree of mutation being more uniformly spread throughout the population.

Thus, the characteristics and possible advantages of repeated mutations are:

- Greater variation in the degree of mutation an individual may experience. This may help the search jump out of sub-optimal solutions, i.e., where a large movement is required to get to a better position.

The characteristics and possible advantages of mutating once are:

- The possible range in which an individual can mutate will be smaller. This could be beneficial for fine-tuning, where small changes and greater stability are required. The degree of mutation, as determined by the mutation rates, will also be more consistently spread throughout the population. This could be beneficial in that more individuals will be changed through the mutations, rather than a few being mutated to a larger degree. As a result a greater number of new points in the solution space will be searched.

5.2.2.1.3 Choosing and Adjusting Mutation Rates

The choice of mutation rates can have a large impact on the effectiveness of an evolutionary algorithm. No single setting is appropriate for all problems, since the effectiveness of a particular mutation rate will vary according to the characteristics of the problem's solution space. For example, a rugged solution surface will require a different mutation rate to that of a smooth solution surface in order to effectively find optimal solutions. Should the solution surface be rugged, a small mutation rate is more likely to get the search process stuck in a local optimum. A larger mutation rate can, in such cases, provide the search process with the ability to jump out of local optimums. However, once a good optimum is being approached a large mutation rate may hinder the search process's ability to fine-tune the solution. That is, it may overshoot the mark. In this situation a smaller mutation rate would be more beneficial. Thus, the stage of the search process and the specific characteristics of the problem's

solution surface can have a big impact on the effectiveness of a particular mutation rate.

No useful theory has been developed in the field of evolutionary computation to guide the experimenter's selection of mutation rates. Often the choice is left to the experimenter's intuition or empirical experience in the problem domain.

Three methods of choosing and adjusting mutation rates are:

- Empirical testing
- Altering the rate as a function of time
- Self-optimization

5.2.2.1.3.1 Empirical Testing

It may sometimes be necessary to resort to empirical experimentation with various mutation rates - if not to find the optimal, then at least to obtain feedback as to the performance levels that can be expected and used as benchmarks to compare with other methods of setting mutation rates. This is the method we employed to compare our selection and variation methods as well as providing a benchmark for an evaluation of our self-optimizing method, where the evolution model optimizes its own mutation rate parameters (see appendices A, B and C for complete data regarding the performance of our selection, variation and self-optimization methods).

5.2.2.1.3.2 Altering Mutation Rates as a Function of Time

Some strategies advocate starting with a large mutation rate and decreasing it as a function of the simulation time. The initially high mutation rate ensures that large areas of the solution space can be quickly sampled. As the mutation rate decreases and the population converges towards an optimum, the ability of the search to fine-tune is improved.

Michalewicz [Michalewicz, 1996] presents such a non-uniform mutation operator that depends on the amount of simulation time that has elapsed.

It is defined as follows: for a parent chromosome $X = \langle x_1,\ldots,x_n\rangle$, if the element x_k is selected for mutation then the result is $X' = \langle x_1,\ldots,x'_k,\ldots,x_n\rangle$, where

$$x'_k = \begin{cases} x_k + \Delta(t, u_k - x_k) & \text{if a random binary digit is 0} \\ x_k + \Delta(t, x_k - l_k) & \text{if a random binary digit is 1} \end{cases}$$

where the domain of x_k is $[l_k, u_k]$ and the function $\Delta(t,y)$ returns a value in the range $[0,y]$ such that the probability of $\Delta(t,y)$ being close to 0 increases as t increases. Thus, smaller changes occur as the simulation progresses.

The function Δ(t,y) used by Michalewicz makes use of the maximum generation number, i.e., the length of the simulation, to produce the random numbers. As t approaches this value, the random numbers that are generated by Δ(t,y) have a greater probability of being near 0. The operator is non-uniform as initially the solution space is uniformly searched, similar to conventional mutation. However, as time progresses the search becomes progressively more locally focused due to the increasingly non-uniform nature of the random numbers (increasingly biased towards 0).

This is also similar to the process advocated in simulated annealing. Neural network Boltzman machines make use of simulated annealing to provide a degree of decreasing randomness to the optimization process. The randomness provides the process with the chance to escape from local optima. As it is decreased the process can be fine-tuned. Other neural network learning algorithms also advocate reducing the learning rate as time goes by.

Problems with such techniques occur when it is not known beforehand how long the training period should be or how many generations are required to evolve an optimal solution, i.e., how quickly the learning or mutation rates should be reduced.

It may also not be appropriate to consistently decrease the learning or mutation rates. Periods requiring higher rates could follow stages utilizing lower rates, depending on the terrain of the solution space that is being covered.

Lastly, situations may occur where continuous learning or evolution is required to adapt to new situations, i.e., the terrain of the solution space may change. There could be no clearly defined training period, as the situation being trained for would change.

A better form of online adjustment is required. A method is needed that can take advantage of current feedback regarding the particular stage that the search process may be in, and set the mutation rate appropriately.

5.2.2.1.3.3 Self-optimization

An approach of simulating evolution known as evolution strategies was the first to incorporate the idea of self-optimization, which is the ability of the algorithm to optimize its own parameters. Instead of a global mutation parameter, the individuals in the population include their probability of mutation as a component of their own description. The individual's mutation rate can then be optimized by the evolution strategy just like any other attribute defining the solution.

Evolution strategies describe an individual as a vector pair ($\mathbf{x}$, $\boldsymbol{\sigma}$), where $\mathbf{x}$ is a vector of n dimensions describing the position in the solution space that defines the solution and $\boldsymbol{\sigma}$ is a vector of variances that correspond to each dimension of $\mathbf{x}$ and is used to mutate the solution.

A solution can then be mutated according to:

$$x'_i = x_i + N(0, \sigma_i)$$

where i = 1,…,n and N(0, σ) represents a Gaussian random variable with a mean of 0 and standard deviation σ.

The vector of variances used for mutating **x** can itself be mutated according to:

$$\sigma_i = \sigma_i .\exp (N(0, \Delta\sigma))$$

where Δσ is a parameter of the method.

Thus, individuals contain the degree to which they are mutated (the vector of variances) as an aspect of their definition. Thus, their degree of mutation is available for optimization by the evolution strategy.

5.2.2.1.3.3.1 Self-optimizing our Agent's Mutation Rate

Our agents' individual mutation rates are made available for optimization by incorporating them into their definitions. The same cannot be done to the population mutation rate as it concerns the population as a whole, which is not a unit of selection. Thus, the population mutation rate must be manually set.

The individual mutation rate, though, concerns only the individual agent, which is a unit of selection. By making the individual mutation rate a part of the agent's description, it becomes visible to selection and, thus, optimization. Although the agent as a whole is selected, its fitness is in part dependent on its mutation rate. Beneficial mutation rates will help agents maximize their survival chances, thus those agents that have the appropriate rates will come to dominate the population.

Should an agent be selected for mutation, according to the population mutation rate, then the change in our agent's individual mutation rate is determined according to:

$$\Delta\text{mutation rate} = \begin{cases} x & \text{if a random digit is 0} \\ 0 & \text{if the random digit is 1} \\ -x & \text{if the random digit is 2} \end{cases}$$

where x is a small, constant value and the random digit is produced in the range [0..2].

Values for x of 0.025, 0.05, 0.075 and 0.1 were examined, with the best results produced using 0.05 (see appendix B for details).

5.2.1.3.3.1.1 Optimization Starting With an Initially High Mutation Rate

As was previously suggested in section 5.2.1.3.2 some methods advocate starting with a high mutation rate and decreasing it over time. This provides the search with the initial ability to quickly sample large areas of the search space and then gradually move towards fine-tuning the search. We also applied this strategy to our optimization technique (the results of the previous section utilized an initial midpoint mutation rate of 0.5). Experiments were also conducted using initial starting rates of 0.1 and 1 as well as randomising all the agent's mutation rates in the initial population between 0.1 and 1. These experiments confirmed the benefits of optimization starting with an initially high mutation rate (a rate of 0.5 giving the best results). The complete data for these results is provided in appendix C. The following graph (Figure 5-4) represents the progression of the fittest and average individual's mutation rate as it optimizes starting from an initial mutation rate of 0.5 for all individuals and a population mutation rate of 1.

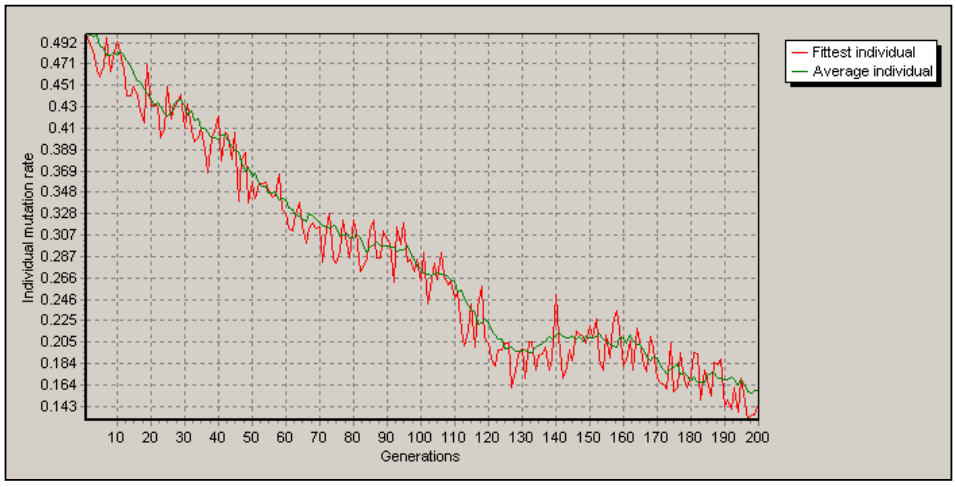


Figure 5-4 Self-optimization of the individual's mutation rate starting from an initial rate of 0.5.

Clearly this graph shows that there is a directed progression towards a smaller mutation rate (0.1589 being the last recorded average individual's mutation rate) as the simulation proceeds. This indicates the ability of the model to move the individual's mutation rate towards more optimal values. As previously mentioned, experiments starting with small mutation rates of 0.1 did not produce superior results, thereby indicating that the higher initial mutation rate does indeed allow the search to quickly sample large areas of the search space with optimization fine-tuning the search as the simulation progresses.

It may be possible to choose a fixed mutation rate that will produce better results, although not by much, as our exhaustive empirical experimentation revealed. However, self-optimization can remove the tedious time consuming effort of empirical experimentation and, more importantly, provides a greater degree of adaptation towards changes in the environment, since self-optimization is not biased

towards a particular environment as our empirical experiments to find a fixed mutation rate were. This is important to consider when dynamic environments are being dealt with. The results from experiments that provide a comparison between a fixed mutation rate and a rate that is allowed to self-optimize in an environment of changing stability will be presented in a following section providing an evaluation of our model's performance in such environments.

5.2.2.1.4 Why Two Mutation Parameters?

Conventional systems use one mutation rate to determine the degree of mutation that should occur in the population. For example, a classical genetic algorithm, representing individuals as bit strings, could assign an equal probability of mutation to each bit. This would be the mutation rate.

Such a mutation rate, then, is evenly spread over all the units of mutation (the bits) in the population. Thus, all the individuals in the population will experience the same probability of mutation, given that they are composed of the same number of bits. Our system for evolving agent neural networks makes use of two mutation rates: a population level mutation rate and an individual level mutation rate. This allows us to specify an agent's probability of being chosen for mutation and if chosen its degree of mutation. Or more specifically it allows us to specify the degree of mutation that will occur in the population as well as the degree of mutation that will occur in the chosen individuals. This gives us more control over the nature of the mutations that the population will experience. We can create situations where all the individuals will definitely mutate, but only to a small degree or we can design the simulation such that only a few individuals will mutate, but when they do it will be to a large degree. Thus, we can create the effect of concentrating mutations in some individuals while leaving others untouched.

The benefits can be related to the concept of elitist selection. Elitist selection is a method to ensure that the fittest individual in a population of solutions is not lost due to changes like crossover or mutations. The fittest individual in a generation is compared to the fittest individual yet found. If it is superior then it becomes the elite individual and is kept until a fitter individual is found.

Our approach does make use of an elitist type selection method to reduce the possibility of an effective individual being adversely affected by mutations. However, this is only applied to a single individual, the individual with the highest fitness in a single generation.

A better mechanism may be required to ensure that useful knowledge, encoded in the form of the fitter individuals, is not lost.

Using two mutation parameters can provide such a mechanism. The population mutation rate can be used to ensure that only a certain percentage of the new population, produced from the fitter individuals of the previous generation, is available for mutation. The remaining percentage is left unaltered. For example, a population mutation rate of 0.5 will result in 50% of the population being mutated and the other 50% being left unaltered. Thus, a new population, that has been created

through selection of the fittest individuals from the previous generation, will have 50% of its individuals changed, to a degree determined by the individual mutation rate, and 50% are guaranteed to stay the same. The result is less risk that knowledge will be lost due to adverse mutations.
This can be related to the concepts of exploration versus exploitation associated with reinforcement learning. An agent seeking to maximize its positive reinforcement needs to find the optimal balance between searching for new ways to improve its abilities and exploiting previous solutions already found. If it focuses on exploiting a particular solution then it runs the risk of missing potentially better solutions that may exist. If it focuses on always searching for new solutions then it may forgo the benefits to be gained from exploiting solutions already found.

By providing an extra mutation parameter we allow a percentage of the population to exploit their position and we force another percentage to explore the solution space. In addition, ensuring that a certain percentage of the population is left unaltered allows us to take greater risks with the percentage of the population being used for exploration. This means that we have a greater degree of freedom in selecting individual mutation rates. Mutating individuals by large amounts can be done with more safety and potentially greater rewards than that which would occur using the conventional single mutation parameter. Our results reinforce this conclusion by indicating that distributing mutations evenly amongst all the individuals in the population (i.e., using a population mutation rate of 1) is not as useful as mutating a percentage of the population and leaving another percentage unaltered. Our results for the selection and variation methods as well as self-optimization consistently indicate that more effective performances are often given by a population mutation rate that is high, but less than 1. To emphasize this fact we draw attention to the data produced by our barometer selection method with single mutations presented in appendix A and the same strategy utilizing self-optimization, with the effective starting mutation rate of 0.5 and different degrees of change, presented in appendix B. As will be later shown, these results were produced from our most successful combination of strategies and indicate that the most effective performances are produced using a population mutation rate between 0.5 and 1. As a result of this data a population mutation rate of 0.8 is chosen for all further experiments.

5.2.3 Performance Comparison of Our Selection and Variation Methods

Experiments were conducted using our two selection methods combined with our two variation methods over a range of population and individual mutation rates that contained the most effective results (see appendix A for complete data).

Three measures of effectiveness were used.

- Maximum average fitness – the maximum average fitness reached in a generation by all the agents in the population for the results averaged over 30 trials.
- Averaged maximum average fitness - the average of the maximum average fitness levels reached by the agents in each trial over 200 generations.
- Average fitness area- the area of the average fitness scores over 200 generations and averaged over 30 trials.

Average fitness scores are preferred since there is a greater degree of variation involved with the maximum fitness scores that are produced by single agents in a generation as a single agent's performance may be largely influenced by the environment it finds itself in. Using the average of all the agents' fitness scores reduces the influence of any unusual environmental circumstances on the performance measure.

The following are the best results produced by individual samples for our selection and variation methods (Figure 5-5).

	Maximum average fitness	Averaged maximum average fitness reached	Average fitness area
Barometer – mutate once	9.51	10.8	1376
Barometer – repeated mutations	9.32	10.69	1331
Roulette – mutate once	8.91	10.21	1249
Roulette – repeated mutations	8.81	10.29	1237

Figure 5-5 Best sample results utilizing barometer and roulette selection methods with single and repeated mutations.

The barometer selection method produced superior results using both variation methods than those of the roulette selection method. Single mutations produced superior results using both selection methods, except for one case, the roulette selection method and repeated mutations produced a slightly better score of 10.29 for the average maximum average fitness reached than the 10.21 produced by the same selection method using single mutations.
The following are the average results produced by all samples taken for our selection and variation methods (Figure 5-6).

	Maximum average fitness	Averaged maximum average fitness reached	Average fitness area
Barometer – mutate once	8.45	9.78	1173
Barometer – repeated mutations	8.37	9.65	1149
Roulette – mutate once	7.64	9.05	1044
Roulette – repeated mutations	7.53	8.93	1022

Figure 5-6 Average sample results utilizing barometer and roulette selection methods with single and repeated mutations.

The conclusions that can be reached from these results are consistent with those derived from the previous results.

The barometer selection method once again produced superior results using both variation methods than those produce by the roulette selection method. These results, though, indicate that single mutations are superior to repeated mutations for all three performance measures, since single mutations produced superior results, for all measures, using both selection methods to those of repeated mutations.

Thus, we are led to the conclusion that the barometer selection method is superior to roulette selection and single mutations are superior to repeated mutations, at least for our example environment. Therefore, the barometer selection method with single mutations will be used for all further experiments.

These results can be explained as being due to the barometer selection methods placing a greater emphasis on fitter individuals, the fittest individuals being guaranteed to reproduce, with the individuals at the bottom of the fitness barometer having no chance. This is not the case with the roulette selection method where the fittest have the highest, but never a certain chance of reproducing, since even though they may be the fittest they can never have all the space on the roulette wheel. This does not necessarily imply that a barometer style selection method will always be better than a roulette selection method, since different solution space characteristics could produce different results. It may be beneficial to provide unfit individuals with a greater chance of reproducing in order to maintain a greater degree of variation amongst the solutions. However, for our purposes the barometer selection method was shown to be the most effective in the given environment and is, therefore, the method of choice.

Our explanation for the result regarding the superiority of single mutations over repeated mutations is that single mutations provide a greater degree of precision in the application of the mutation rate. When individuals are selected for mutation they are always mutated to a degree that is proportional to the mutation rate. In the case of repeated mutations an individual may be mutated several times. Thus, the degree to which they may be mutated might vary over a large range (several times the individual mutation rate). Ensuring that an individual is only mutated once allows for the mutation rate to be evenly applied to a specific percentage of the population. For example, if the population mutation rate is 0.5 then it is assured that 50% of the population will be mutated, as compared to a lower percentage being mutated to a larger degree. Thus, the population and individual mutation parameters can be applied with a greater degree of precision, since we can be certain of the degree to which an individual will mutate and the number of individuals that will be selected for mutation.

In addition, in the case of self-optimization, the greater degree of precision in applying the mutation rates provides more informative feedback to the evolution model, i.e., with regard to what individual rate is associated with higher fitness scores. This means that the individual's mutation rate can be more easily optimized. Repeated mutations are less informative since the individual's mutation rate has a less accurate relationship with the actual degree of mutation experienced by that individual.

5.2.4 Our Evolution Model

To conclude this section we present the final attributes of our evolution model, as influenced by the above results.

The general algorithm is summarized as follows:

```
t=0;
//initialise the population of agents
Initialise P (t);
Repeat
  //Allow the agents to interact with the environment for
  //a certain number of epochs (one generation)
  Simulate environment;
  //evaluate the fitness of the agents in the population
  Evaluate P (t);
  //create a new population based on the previous population's agents' fitness scores
  //using an elitist strategy and the probabilistic barometer selection method
  P (t)′ = reproduce P (t)
  //provide variation (new agents) through mutation
  //using selection without replacement and two mutation parameters
  //in the case of self-optimization, the individual mutation rate is
  //included in the description of the agent
  P (t+1) = mutate P (t)′ //except the previously fittest (elitist) individual
  //increment the generation count
  t=t+1;
Until a set number of generations have passed
```

5.2.4.1 Initialization

At initialisation each agent's parameters are set to appropriate starting values (i.e., fitness = 0). If the individual mutation rate has been selected for optimization then it is either randomised between 0.1 and 1 or it is set to a specified starting value.
The agent's neural network is initialised by randomly selecting a starting weight of -1, 0 or 1 for each of its connections.
These measures ensure that the initial population provides a number of different starting points within the solution space.

5.2.4.2 Simulating the Environment

The agents interact with the environment for a specified number of epochs. During this time they can navigate around their environment by using their abilities to move forward and rotate in both directions. These actions are chosen by an agent's neural network in response to inputs from its visual system.

The agents come into contact with two other objects that exist in their world, food and poison. Consuming them occurs if the agent makes contact, with appropriate consequences to the agent's fitness.

5.2.4.3 Evaluation

After a generation has passed the agents are evaluated according to the following fitness function:

$$Fitness = x.f + y.p$$

where x is the amount of food eaten, f is the fitness consequence for eating food, y is the amount of poison eaten, p is the fitness consequence for eating poison.

5.2.4.4 Reproduction

The fittest individual is always reproduced using an elitist type selection method (section 4.2.1.3). The barometer selection method (section 5.2.1.2.2) is then utilized to provide each agent with a probability of reproducing in proportion to its fitness scores. The algorithm iterates over the population giving each agent a chance to reproduce. It repeats the loop until the new population has the required number of members.

5.2.4.5 Mutation

To create new knowledge, the population is mutated according to two mutation parameters. Firstly, the population mutation parameter determines the number of individuals to be mutated. Individuals are randomly selected, except for the elitist individual, since we want it to stay the same.
On being selected the individual mutation rate is used to mutate the individual. This rate may be a parameter to the system or it may exist within the agent, in the case of self-optimization.
The individual mutation rate influences the number of neural network connections that must be mutated. Connections are randomly selected and mutated to a specified degree.

In the case of both the population mutation rate and the individual mutation rate, individuals and their neural network connections can only be selected once for mutation, since it was found to be more effective than repeated mutations (section 5.2.3).

Once the new population has been created the process is repeated until a specified number of generations have elapsed.

5.3 Evaluation of Our Evolution Model

The evaluation of our evolution and reinforcement learning models are approached in the same way. This involves investigating the performance and adaptive nature of each model in an environment that ranges from stable to highly unstable, thereby allowing us to draw conclusions regarding the adaptive abilities of each model towards stable environments and changes in such environments.

5.3.1 Performance in Stable to Unstable Environments

5.3.1.1 Overview

The following levels of environmental stability were investigated, with each value representing the number of generations that elapse before a fitness consequence alternation occurs: 200, 100, 50, 25, 13, 7, 4, 2, 1. Since our simulations run for 200 generations, the first environment shows no changes. Thus, our models are investigated in environments that range from stable to highly unstable, utilizing fitness consequence alternations after every generation.

Each model is evaluated over 30 trials in the environment utilizing a particular stability level, with three fitness values being recorded to measure the models' performance. Two of these are with regard to the maximum average fitness levels produced by the agents and a third measures the sum (area) of the average fitness levels over the course of the simulation. This last measure is utilized for the following and other evaluations, since it is the most informative with regard to consistency of performance. However the data using all three measures point to the same conclusions and are available in appendix D.

The following graph (Figure 5-7) presents the average fitness area scores of the agents produced by our evolution model over 200 generations and averaged after 30 trials, using a fixed individual mutation rate of 0.1 and a population mutation rate of 0.8.

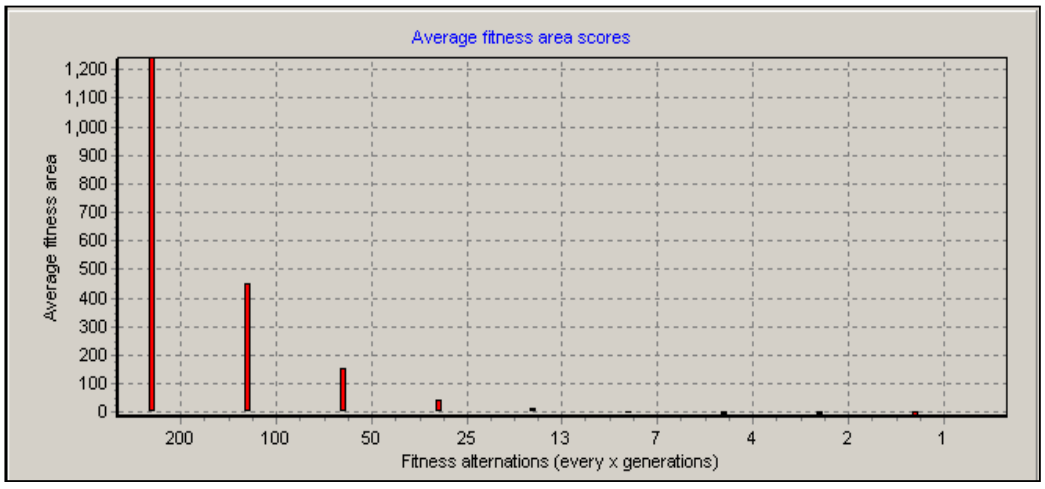


Figure 5-7 Average fitness area scores produced by our evolution model in stable to highly unstable environments.

As is shown by the above graph, our evolution model's performance rapidly degrades with increasing instability, ultimately producing negative fitness scores for fitness consequence alternations beyond every 13 generations. Utilizing self-optimization, as previously mentioned, proved to produce no improvements in performance (see appendix D for details).

To investigate the nature of this performance further we provide details concerning the model's performance in response to several different degrees of environmental stability.

5.3.1.2 Performance Details

5.3.1.2.1 Stable Environment

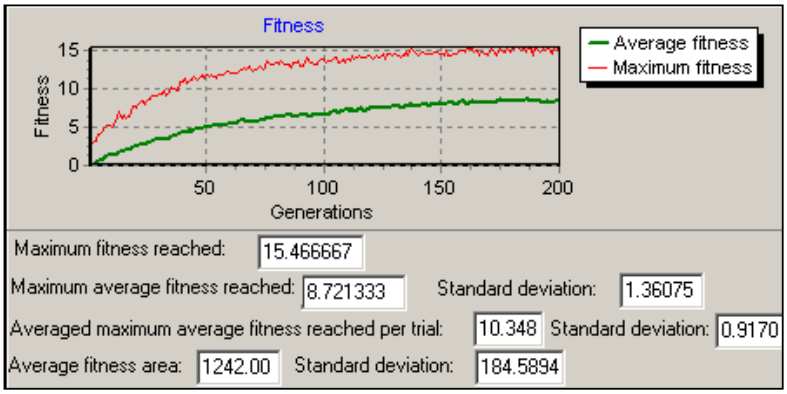


Figure 5-8 The evolution model's performance in the stable environment.

In the stable environment (Figure 5-8), our evolution model demonstrates the ability to steadily progress towards an effective adaptation to the environment. Both the maximum and average fitness measures indicate that the agents come increasingly to know the environment as a result of our evolution model's search.

5.3.1.2.2 Fitness Consequence Alternations Every 25 Generations

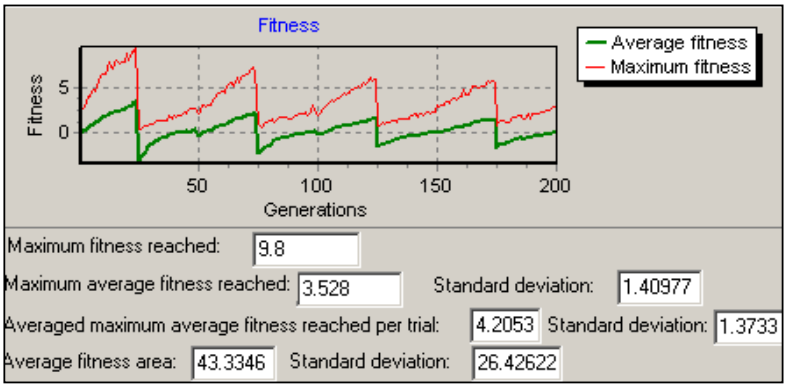


Figure 5-9 The evolution model's performance in an environment with fitness consequence alternations every 25 generations.

Increasing the instability of the environment, by utilizing fitness consequence alternations every 25 generations, results in a markedly different performance. The above graph (Figure 5-9) clearly shows that the evolution model has difficulty adapting to such an environment. Beyond 25 generations no further improvement in fitness scores occur. Thus, the model lacks the ability to incorporate the fitness consequence reversals in order to adapt more effectively to the environment. Any adaptations produced become maladaptations once the fitness consequence reversal occurs. That is, adaptations to eating food become adaptations towards eating poison as soon as the roles of each object are reversed. This explains the oscillating nature of the fitness performances, which only become worse as the level of instability is increased.

5.3.1.2.3 Fitness Consequence Alternations Every 7 Generations

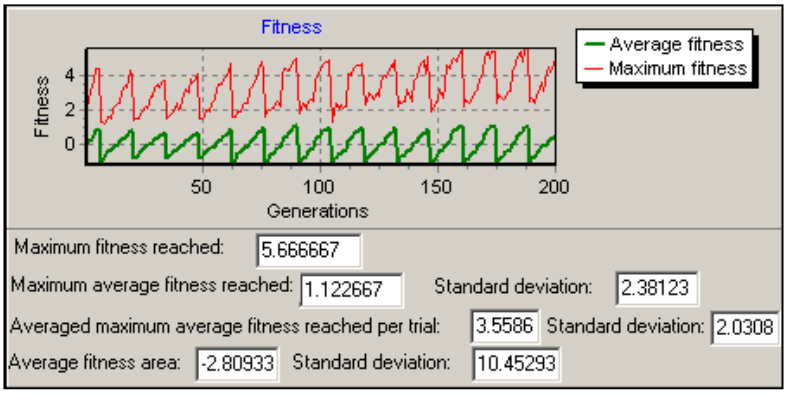


Figure 5-10 The evolution model's performance in an environment with fitness consequence alternations every 7 generations.

Increasing the frequency of the fitness consequence reversals to occur every 7 generations confirm our previous statements regarding the model's adaptive nature. The fitness performances of the above graph (Figure 5-10) demonstrate that the model lacks the ability to incorporate the increasing degree of instability, demonstrating wildly oscillating fitness scores with no improvement in effectiveness over the simulation's 200 generations. In addition the model produces a negative average fitness area result (-2.80933), thereby further indicating the inability of the model to deal with such levels of instability, producing agents that consume more poison than food over the simulation's duration.

5.3.1.2.4 Fitness Consequence Alternations After Every Generation

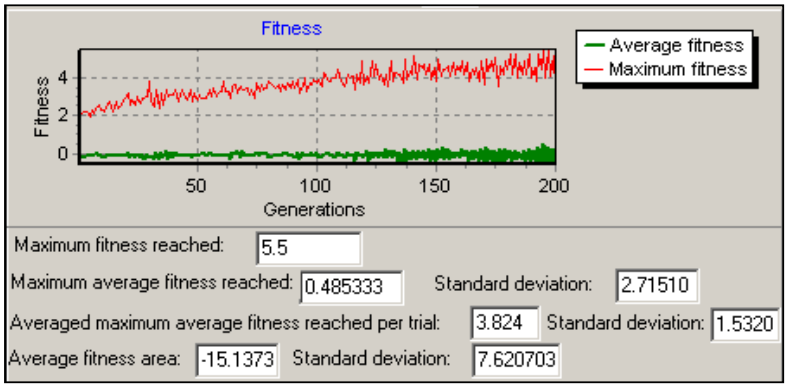


Figure 5-11 The evolution model's performance in an environment with fitness consequence alternations after every generation.

In the environment demonstrating the highest degree of instability, the evolution model's performance is completely ineffective (Figure 5-11). Although slight improvements in fitness peaks do occur, they result from the increasingly oscillating nature of the performance - note the increasing highs and equivalent lows of the average fitness scores as the simulation progresses. These oscillating average fitness levels towards the end of the simulation are due to a strategy of agents eating a particular object and avoiding the other. The increasing amplitude of the fitness oscillations are due to the agents becoming more effective at eating a particular object, regardless of its fitness consequences. This has the result of producing maximum fitness levels that improve over the course of generations, since there is likely to be, in any generation, an agent that is correctly locating the food object.

However, this response to the increased instability produces an average fitness area that is once again negative (-15.1373), indicating that the agents are consuming more poison than food over the duration of the simulation. Thus, it cannot be considered an effective adaptation.

The following presents an investigation of the adaptive behaviour and knowledge of the agents in order to help us better understand the above results and the adaptive nature of the evolution model.

5.3.2 Adaptive Behaviour and Neural Network Knowledge

Our evolution model produces agents that contain innate, fixed knowledge that allows them to behave effectively in their environment. This is achieved through the processes of selection and variation in the context of the population's experience of the environment. As we have shown, this was effective in producing agents that can adapt to the environment demonstrating stable characteristics. However, after increasing the level of instability beyond a certain point, the model is considered ineffective in dealing with such environments, due to the model's limited ability to adapt beyond a certain rate of change.

In order to demonstrate this further, consider firstly an effective solution in a stable environment.

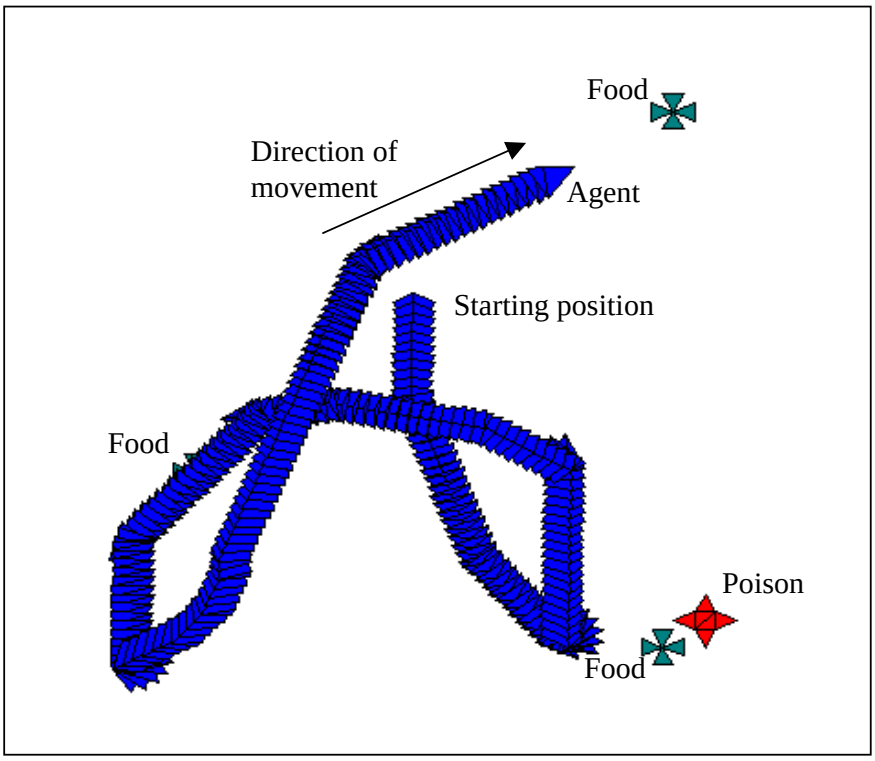


Figure 5-12 A trace of a successfully evolved agent in a stable environment.

The above figure (Figure 5-12) is a trace of an agent effectively navigating its environment in order to locate three food objects. The neural network knowledge required to produce such behaviour is shown in the following two figures.

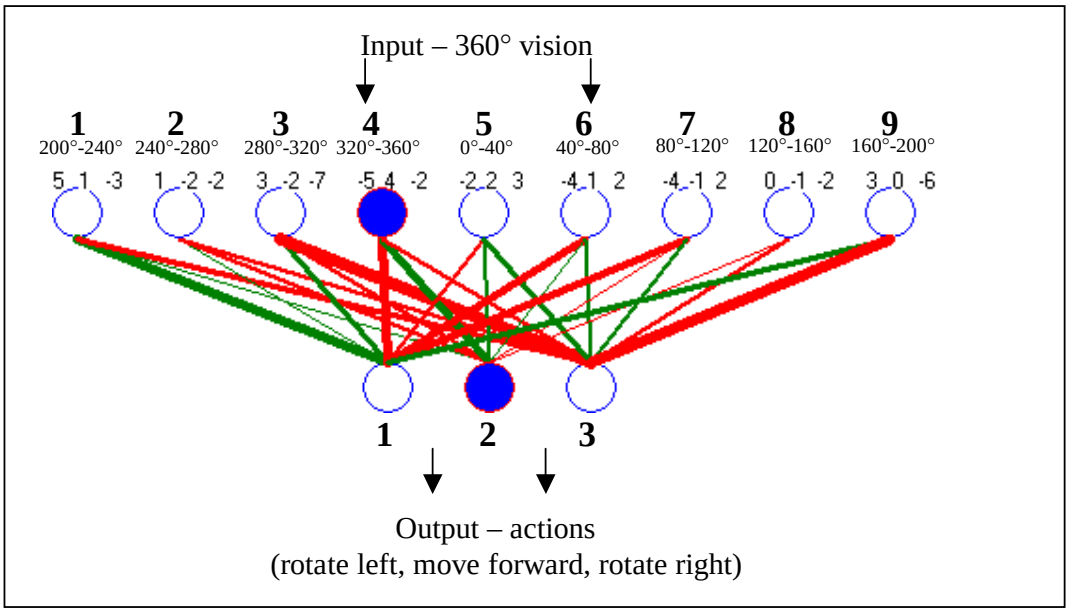


Figure 5-13 A successfully evolved neural network responsible for mapping visual input regarding green objects (food in the stable environment) to an output action.

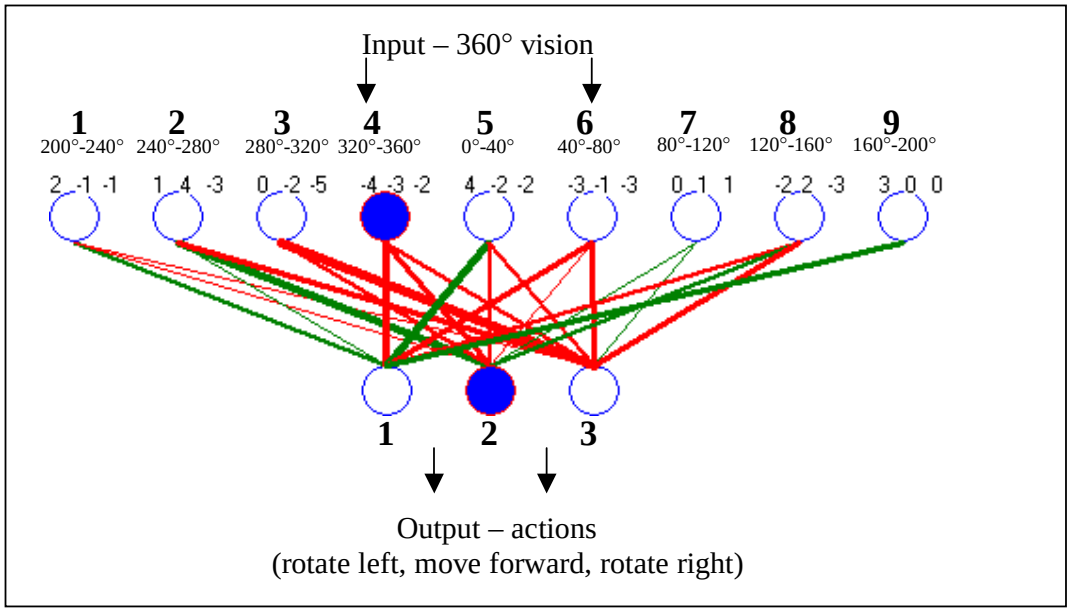


Figure 5-14 A successfully evolved neural network responsible for mapping visual input regarding red objects (poison in the stable environment) to an output action.

Figure 5-13 and 5-14 represent the portions of a successfully evolved agent's neural network responsible for mapping visual input regarding green and red objects (food and poison, respectively, in the stable environment) to an output action. Thus, 9 input neurons and 27 weights are devoted to each object with the bottom output neurons being shared. The chosen action, then, is influenced by the position of both green and red objects.

The colour and thickness of the connections represent their sign and strength. Red is negative (inhibitory) and green positive (excitatory). Their strength is also numerically represented above each input neuron, the order of the values corresponding to the index of the output connection (i.e., the first value above each input neuron corresponds to the strength of the connection to the first output neuron). The visual range that each input neuron is responsible for is also indicated above these values.

Using this information we can extract the agent's knowledge responsible for its adaptive behaviour. The following figures were produced from this information and represent the agent's actions that are most likely to occur, given the position of the objects relative to the agent.

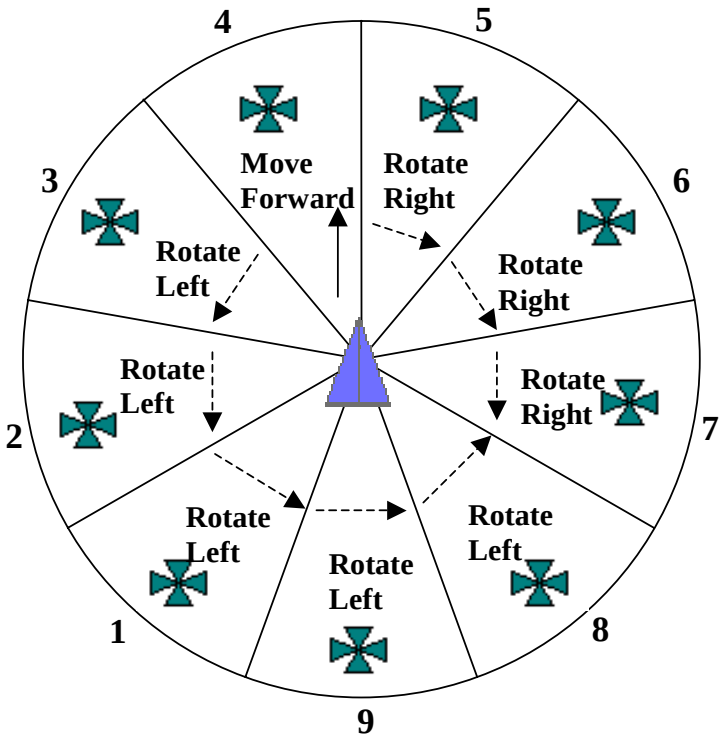


Figure 5-15 The influence of the position of green objects (food in the stable environment) on the agent's behaviour.

Figure 5-15 indicates how the agent comes to orientate itself towards green (food) objects. Note that regions 4 and 5 are directly ahead of the agent, between 320°and 40°. The above diagram indicates that our agent implements a strategy of moving the food object into region 4 before moving forward and, thereby, successfully locating it. That is, if the food object is generally to the right of that region, then the agent will rotate in that direction. Should the food object be to the left of region 4 then the agent rotates in that direction.

The following figure (Figure 5-16) represents the agent's reactions to red (poison) objects. What is apparent from this diagram is that the agent demonstrates an aversion towards poison existing in positions directly ahead of it. If it should occur in region 4, the agent rotates to its right, bringing the poison towards region 3. Should the poison occur in region 5 then the agent performs a leftward rotation bringing it to region 6. What is also apparent from the mappings is that the agent demonstrates a preference towards poison existing in region 2, that is, to its left between 240° and 280°. This can be deduced since the forward movements and leftward rotation mappings of regions 6, 7, 8, 9 and 1 will bring the object to this position. Once it is in region 2, the move forward mappings will help keep it there.

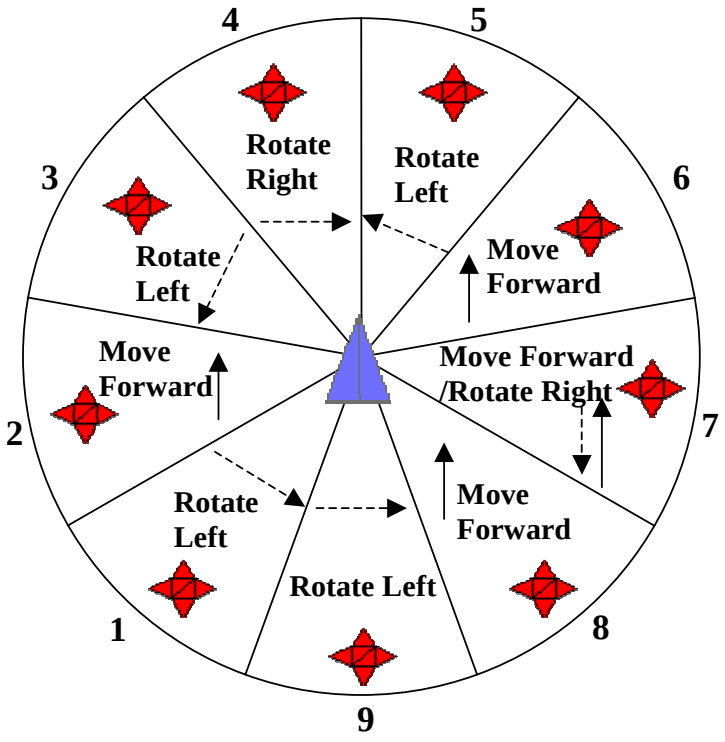


Figure 5-16 The influence of the position of red objects (poison in the stable environment) on the agent's behaviour.

The above description of the agent's behaviour helps us to understand why it is successful in the stable environment, but why is the evolution model not successful in producing agents that are effective under increasing levels of instability?

The evolution model is limited in its ability to effectively adapt to such situations for the following reasons. The problem is of a difficult nature that requires two solutions containing opposite attributes. One solution entails producing behaviour to locate green objects while avoiding red objects. The other solution requires that the agents locate red objects while avoiding green objects. These two solutions, then, are of a conflicting nature, with the requirement that only one should be present in the agent to adapt effectively to the characteristics of the environment. Thus, as the characteristics of the environment change, after a fitness consequence reversal, it is required that the previously implemented solution be discarded for its opposite. This is a time consuming process for the evolution model as it utilizes limited feedback at a population or generation level to direct changes and, in addition, the changes are slow to implement since the solutions are of an opposite nature that do, however, utilize the same portions (memory) of an agent's neural network. Thus, it is required that previous solutions are unlearnt, as opposed to being set aside, while the new, appropriate solution is relearnt. In addition, since feedback only occurs at the end of an individual's lifetime, the incorrect solution is implemented throughout the agent's lifetime without any resistance (i.e., lifetime learning) until a sufficient number of generations have elapsed for it to be unlearnt. As a result, negative fitness scores are recorded until the previous, now incorrect, solution has been rendered ineffective.

This process is repeated each time a fitness consequence reversal occurs and, therefore, makes adaptation and assimilation of such changes difficult for evolution models to effectively deal with.

5.4 Conclusion

Our evolution model proves to be an acceptable means of developing our agent's neural networks in the stable environment. Given limited feedback, effective behaviour is evolved. That is, our agents learn to find food and avoid poison.

However, as the environment becomes more unstable, by alternating with a greater frequency which objects are food and poison, the effectiveness of the model degrades to ultimately be considered ineffective in the most unstable environments.

The model is limited in its ability to adapt to such environments, since agents are evolved containing fixed knowledge via a generation or population level learning process. They are, thus, limited in their ability to unlearn incorrect behaviour and relearn correct behaviour under conditions of change.

In addition, the model demonstrates no ability to assimilate the changes and improve learning times over the course of the simulation.

Thus, our evolution model, although quite effective under stable circumstances, demonstrates no ability to evolve solutions in response to highly unstable conditions.

Chapter 6: Design and Evaluation of a Reinforcement Learning Model

6.1 Introduction

Our reinforcement learning model provides the agents with the ability to learn within their individual lifetimes. By making use of positive and negative reinforcements their behaviour can be modified to achieve the desired objectives, maximize their ability to consume food objects and avoid poison objects.

Reinforcement learning is similar to evolutionary learning in that solutions can be produced by defining what is to be done rather than how to do it. However, reinforcement learning works by altering knowledge within an individual in its lifetime by positive and negative reinforcement, the counterparts of evolution's ability to modify knowledge encoded in the population as a result of reproduction and death.

Thus, the learning is of the same nature as evolution models, except that it can now take place within the individual as compared to evolution or population level learning where knowledge is altered over generations by the processes of reproduction and death.

Instead of fitness measures, individuals now require reinforcement signals with regard to events or states of the environment that relate to their objectives. Typically, in nature, these reinforcement signals would ultimately be derived from fitness consequences, the reinforcement signals being the internalisation of an environment's fitness consequences. For our purposes, we provide reinforcement signals that are of the same nature as the fitness consequences utilized by our evolution model. Instead of increasing or decreasing an agent's fitness for locating food or poison, it receives positive and negative reinforcement signals for the same events. Thus, our evolution and reinforcement learning models utilize information that is of an equivalent nature in order to produce effective agents.

However, they are different in that the evolution model can only utilize fitness scores to evaluate an individual at the end of its lifetime to produce a new generation, whereas our reinforcement learning model has the ability to provide an individual with feedback in relation to particular actions and resultant events and, thereby, alter its behaviour accordingly. Thus, our evolution and reinforcement learning models can be considered similar processes that do, however, operate over different timescales. This allows for an interesting investigation into the effectiveness of similar learning processes that occur at two different levels and the interaction between such mechanisms in their combined forms.

The reinforcement learning model is applied to the same environment and agents as previously used for the evolution model. Like the evolution model, it was designed to produce effective, adaptive behaviour in agents whose objective it is to consume food and avoid poison in a virtual environment. It is evaluated in terms of its performance in the environment demonstrating stable characteristics and in terms of its response to changes in the environment's stability.

Thus, our objectives were to:

- Design a reinforcement learning model with a view towards maximizing its ability to produce adaptive agents in an initially stable environment that becomes increasingly unstable.
- Evaluate the effectiveness and adaptive nature of the model with a view towards extracting the advantages and limitations of such models in dealing with stable environments that become increasingly unstable.

6.2 Design

Our reinforcement learning model was designed to train the agents' neural networks to produce behaviour that maximizes their ability to locate food objects and avoid poison objects by utilizing feedback in the form of reinforcement signals. This feedback is only provided in relation to certain events that relate directly to the ultimate goals, i.e., locating food. Thus, the feedback is of a similar nature to that of the evolution model, where the fitness of the agents is only altered after such events. The agents need to use this feedback to appropriately reinforce, either negatively or positively, the instrumental behaviour that was involved in the event occurring. Since such behaviour may occur a certain amount of time before the event takes place, the model must be capable of reinforcing behaviour that produces delayed consequences.

The design of the model, then, needs to satisfy the following main objectives:

- The agents' neural networks must be capable of utilizing feedback in the form of scalar reinforcement values, associated with particular events, in order to reinforce behaviour (input-output mappings) to make those events more or less likely to occur in the future.
- The model must be capable of reinforcing behaviour that is not immediately involved in the feedback events, but instrumental in producing their occurrence. Thus, it must handle the problem of delayed reinforcement.

6.2.1 Reinforcing Neural Network Behaviour

In order to reinforce our agent's behaviour to accomplish the desired objectives we require a method of reinforcing their neural network's stimulus (input) and resultant action (output) mappings according to their involvement in producing effective behaviour.

Reinforcing input-output pairs is known as associative reinforcement learning [Barto and Anandan, 1985]. In the context of neural networks this can be achieved by using the CRBP (complementary reinforcement backpropagation) algorithm (refer to section 2.3.2) of Ackley and Littman [Ackley and Littman, 1990].

However, conventional CRBP alone is not capable of dealing with delayed reinforcement. In addition, it places an emphasis on reinforcing input/output mappings as compared to general neural network activity, since it makes use of backpropagation, a method originally designed to train multi-layer neural networks using desired input/output pairs. Should agents be used that have multi-layer,

recurrent neural networks, then reinforcing neural network activity could be required, as opposed to merely reinforcing input/output mappings. Agents with such neural networks have a much greater ability to process information internally, as a result of multiple layers and recurrent connections. Their output is not dependent on a direct mapping from their input. Ultimate output may go through many cycles of internal processing. As a result, the latest output may be completely unrelated to the agent's latest input. Rather, it may be a result of inputs that occurred a certain amount of time before the output and a large amount of internal processing. Reinforcing the latest input with the latest output would, thus, be inappropriate.

In addition backpropagation has also been criticized for not being biologically plausible. Biological plausibility is important, since we know what organic neural networks are capable of. See work by Baxter [Baxter, 1992] and Schmidhuber [Schmidhuber, 1989] describing local learning algorithms that are biologicaly plausible.

Our approach advocates using information that is local to each connection and a scalar reinforcement signal, since it is unlikely that biological neural networks make much use of globally available information. The dynamics of biological learning are a more likely result of distributed information locally available to each connection. The only globally available information that our reinforcement learning model makes use of is a scalar reinforcement signal.

Utilizing distributed connection information allows us to produce a method that is both more biologically plausible and orientated towards reinforcing neural network connection activity.

Our design, therefore, is with a view to reinforcing neural activity, as compared to just input/output mappings. However, for this work reinforcing neural network activity is equivalent to reinforcing direct input/output mappings, since our agents have single-layer, feed-forward neural networks. Thus, the approach taken is a movement in the direction of reinforcing neural activity.

The first aspect of our model is to reinforce the agent's neural network behaviour (stimulus/action mappings) that occur in direct relation to feedback provided by scalar reinforcement signals, that is, mappings produced immediately prior to a feedback event.

Actions such as eating food and poison are directly associated with positive and negative feedback, as these events are defined, a priori, to be good or bad. This feedback defines the goals we wish our agents to achieve and is equivalent to the fitness consequences that are utilized by our evolution model.

If stimulus is mapped to an action that results in negative feedback (eating poison) then that input-output mapping should be weakened. If the action resulted in positive feedback (eating food) then the input-output mapping should be strengthened.

Our model utilizes the Hebb rule to reinforce mappings. If two connected neurons are active simultaneously prior to reinforcement occurring, then the strength of the

connection between them is modified by the reinforcement value (positive or negative).

This produces the following weight update:

If X_i and $Y_j > 0$ then
$W_{ij}(new) = W_{ij}(old) + r$

where X_i and Y_j are the activation levels of neurons X_i and Y_j,
W_{ij} is the connection strength between them, and r is a scalar reinforcement value provided in response to an event.

This simply strengthens or weakens the mapping that occurred immediately prior to the reinforcement event.

As an example, the following figure (Figure 6-1) represents an agent about to move forward and eat food. On making contact with the food it is considered to have eaten it and a positive reinforcement will be given to the agent.

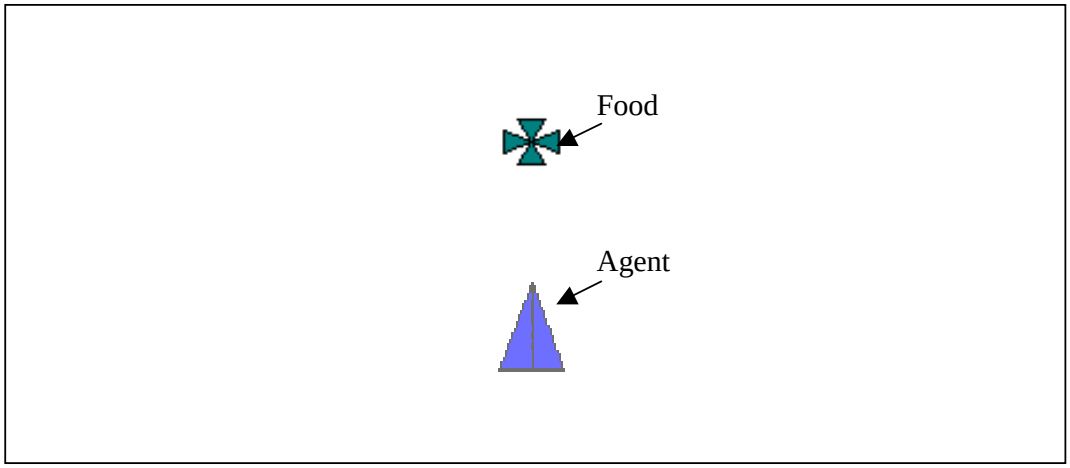


Figure 6-1 An agent about to receive positive feedback.

The figure below (Figure 6-2) represents the agent's neural network activity and connection strengths corresponding to the above figure, before reinforcement occurs. Assume that the mapping to the neuron responsible for generating a move forward action is fortunate, as no knowledge has yet been encoded into the network.

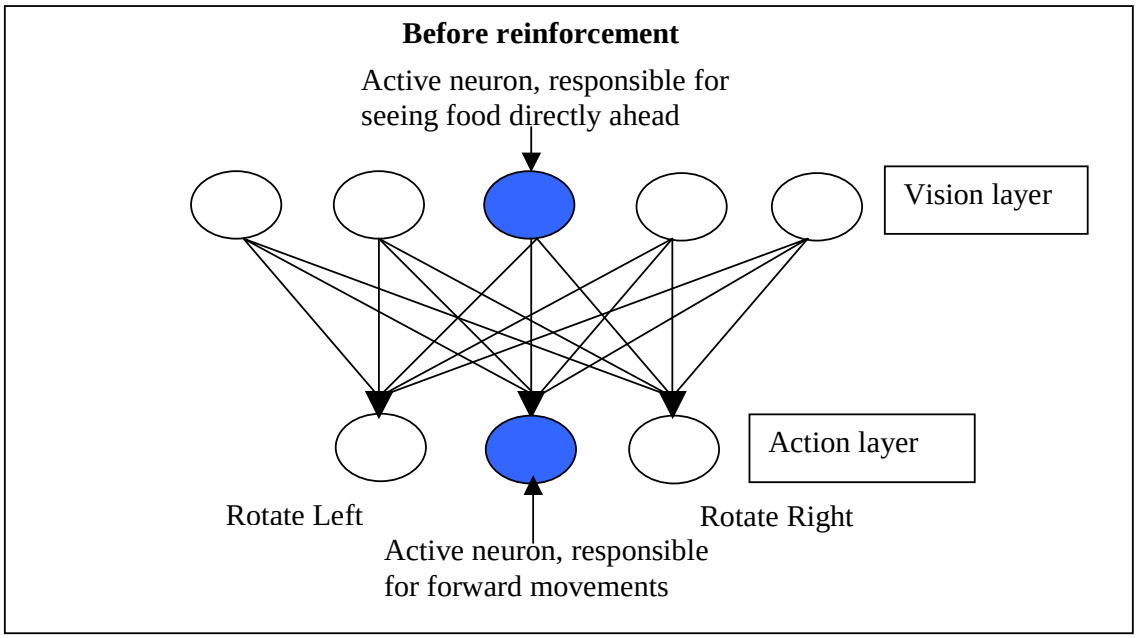


Figure 6-2 An agent's neural network before reinforcement.

Since the agent ate the food, the previous mapping will be reinforced, making the behaviour more likely to occur in the future, as indicated in the next diagram (Figure 6-3).

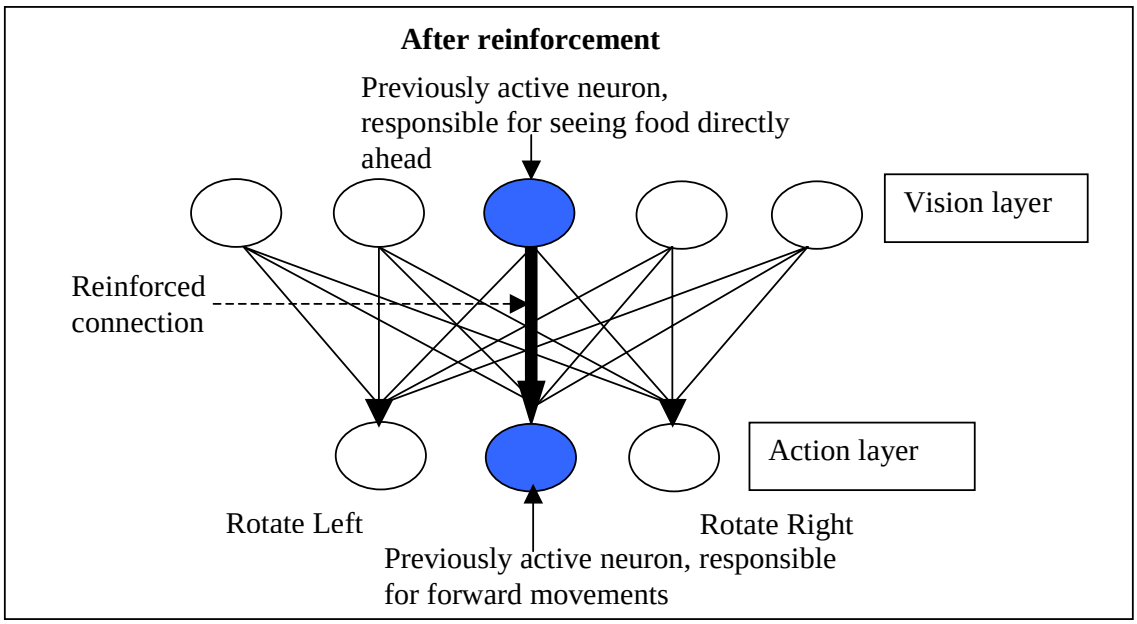


Figure 6-3 An agent's neural network after reinforcement.

The connection's strength between the two previously active neurons has been increased by the reinforcement signal. Should poison have been eaten a negative reinforcement would have occurred, decreasing the connection's weight.

The problem is that the agents need to reinforce the behaviour that got them to the food or poison in the first place. Extracting this behaviour is a problem of delayed reinforcement.

5.2.2 Dealing with delayed reinforcement

In order to extract the instrumental behaviour that was involved prior to the reinforcement event, the agents need to have a method of apportioning the received feedback to previous input/output mappings. For this they need some form of memory that will enable them to extract the stimulus/action mappings that were good for them a long time before the reinforcement event actually took place.

An eligibility trace is the conventional method of dealing with this problem (as discussed earlier, section 2.2.3). We implement the eligibility trace by remembering the time a connection was used between two active neurons. This allows us to place a time stamp on previous connection activities and, thereby, give us an idea of previous neural network activity.

The longer ago a connection was effectively used before a feedback event, the less likely it contributed to that event occurring. Therefore, connections are reinforced in inverse proportion to the amount of time that has elapsed.

The reinforcement function becomes:

$$W_{ij}\ (\text{new}) = W_{ij}\ (\text{old}) + r/(\text{elapsed time} \times \text{decay rate})$$

where W_{ij} is the weight of the connection between neurons X_i and Y_j that were active at the time of the time stamp, r is the delayed reinforcement, the elapsed time is equal to the current time less the time stamp, and the decay rate influences how fast the eligibility trace decays.

A reinforcement event will cause this function to be applied to every connection in the neural network.

Our eligibility trace can be termed a replacing eligibility trace (described earlier, section 2.2.3), as the amount of reinforcement that can be received after a feedback event will have a fixed maximum due to the elapsed time having a minimum of 1. On being reset (every time the connection is used) the trace will always start from this maximum level (elapsed time = 1). Therefore, the number of times the connection is used prior to the feedback event will not affect the amount of reinforcement it receives. This is only affected by how recently it was used.

As an example of our method, consider the following events presented in the figure below (Figure 6-4).

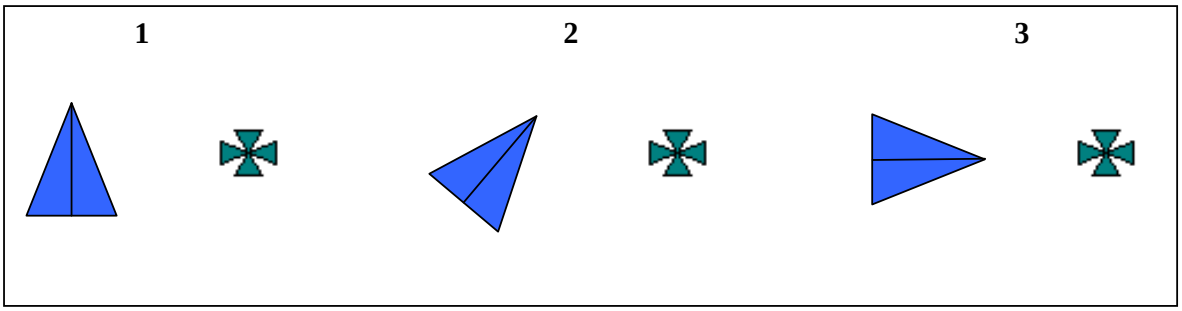


Figure 6-4 An agent coming into contact with food.

This would correspond to the following neural network activations in figure 6-5.

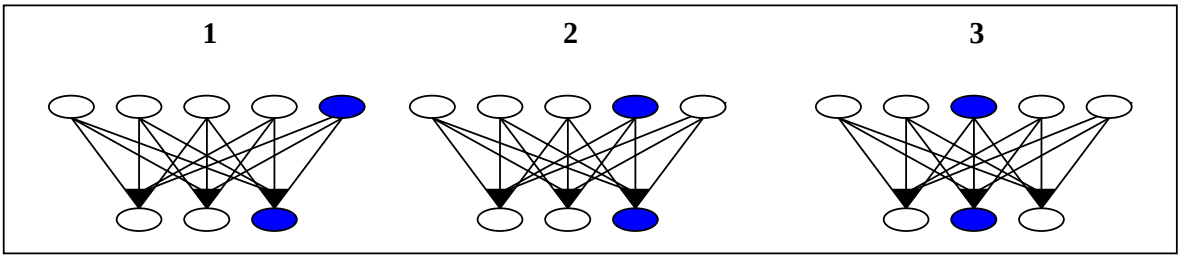


Figure 6-5 Neural network mappings for food vision to actions.

Assume the agent's 5 input neurons are responsible for its frontal vision (270° to 90°).

As the agent rotates towards the food, due to its action neuron responsible for clockwise rotations being activated, its neural network input activations shift from the right to the centre neuron. Directly facing the food it moves forward.

On eating the food a feedback event occurs and the neural network's connections are reinforced relative to the time they were last used, given in Figure 6-6 below (connections not used in the above events are not shown).

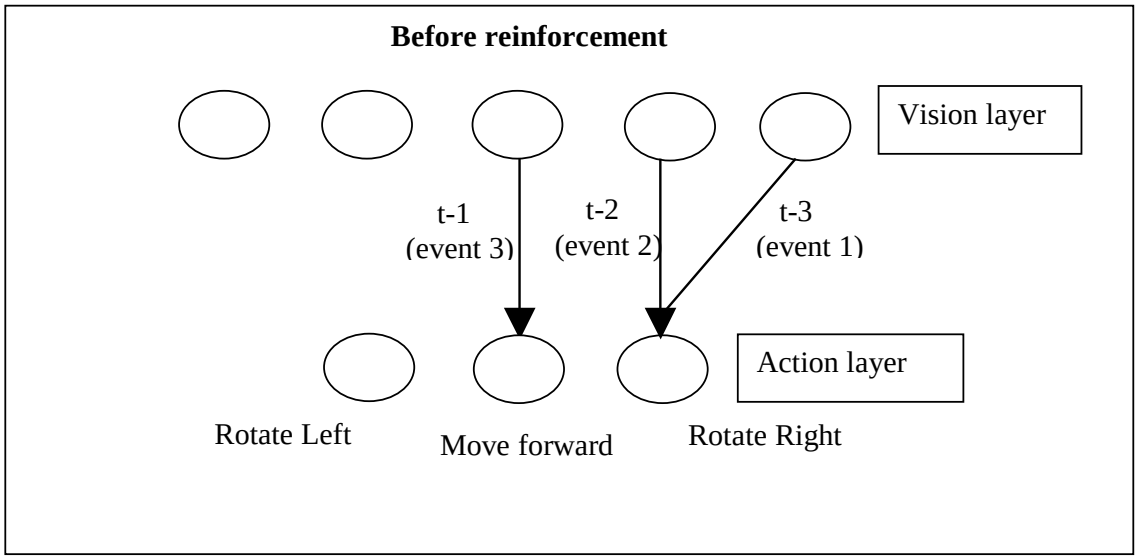


Figure 6-6 The agent's neural network before reinforcement.

Assume that the connection strengths prior to the reinforcement are all equal to 1 (thus, the above sequence of events was fortunate as all the output neurons received the same amount of input resulting in all actions having an equal probability of occurring). Also assume a positive reinforcement of 5 for eating food and a decay rate of 1.

The connections will be updated as follows. The connection involved in event 1 will receive an increase of 5 (the positive reinforcement value)/3 (the elapsed time prior to the feedback event), giving 1.666. The connection involved in event 2 will receive 5/2, giving 2.5 and the connection involved in event 3 will receive an increase of 5, the full reinforcement value since it occurred immediately prior to the feedback event.

The neural network after reinforcement is represented below (Figure 6-7), with connection strengths being indicated by the thickness of the connections. The values of the altered weights are represented next to each connection. All other connections are not utilized and remain with weights of 1.

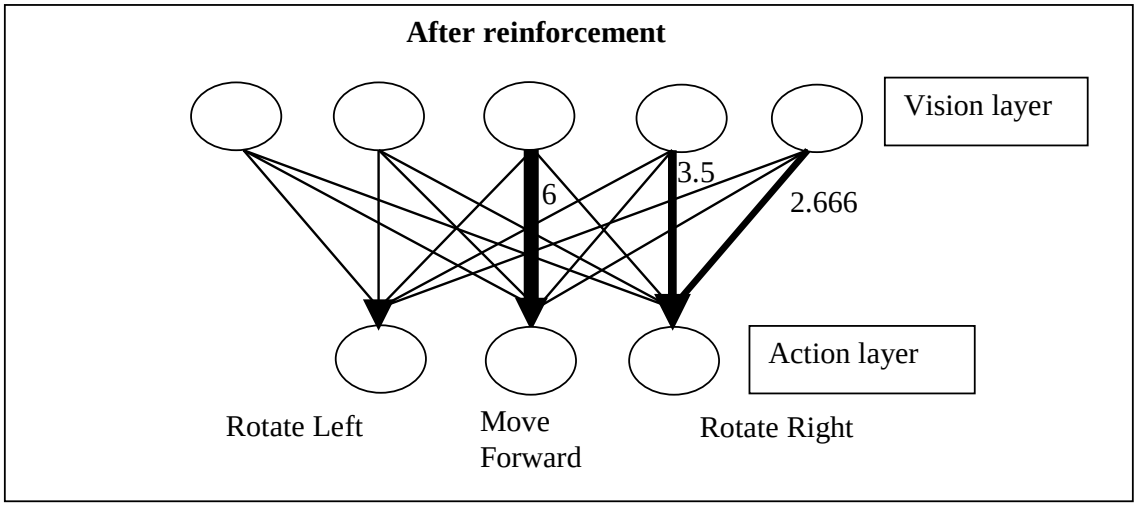


Figure 6-7 The agent's neural network after reinforcement.

The network will now likely reproduce the above behaviour, represented by events 1, 2 and 3 should the agent find itself in the same or a similar situation.

6.2.3 Feedback

Our agents receive positive feedback for eating food, negative feedback for eating poison and a small constant, negative feedback for every other action.

The constant, negative reinforcement per action gives the agent the ability to unlearn previous mappings that may have been unjustly reinforced. This ensures that all actions that do not lead to food being eaten are negatively reinforced. This means that each action has a "cost". To be "profitable" their actions must result in the consumption of food to the extent that the positive reinforcements received are greater than the constant, negative reinforcement per action. This has the effect of optimising behaviour and, thereby, producing a more effective agent. Our evolution model naturally optimized behaviour, as a result of competition.

This feedback also helps the agent try something else should its behaviour result in it being stuck in a difficult position.

These three reinforcement parameters, then, communicate our objective to the agents, locate food and avoid poison.

6.2.4 Simulation Framework and Parameters

We use the same simulation framework utilized for our evolution model to perform empirical testing of various feedback values.

Important constants, as used to examine our evolution model, are:

- A generation lifetime of 500
- A simulation length of 200 generations
- A population of 25
- 1 food object in the world at any time
- 1 poison object in the world at any time
- 30 trials per sample

Performance results are still presented with regard to average fitness results achieved by the agents within each generation. A population of 25, as used for the evolution model, provides these fitness results. Note, that a population is not a necessary factor of the reinforcement model. Rather, it is used to provide significant average fitness results for comparison with the other models.

At the completion of each generation the same agents are used for the following generation, with their fitness scores being reset to 0. They keep their learnt knowledge, since this is equivalent to the agents in our evolution model inheriting the knowledge of their parents.

Empirical experiments were conducted to find good values for the reinforcement parameters. We use equal, but opposite feedback values for our reinforcement model's feedback for eating food and poison, since our experiments with the evolution model used equal, but opposite fitness consequences with regard to the same events. The small constant, negative feedback value used for every stimulus/action mapping does not have a direct equivalent in our evolution model, since that model's population based competition provides the necessary optimization pressure.

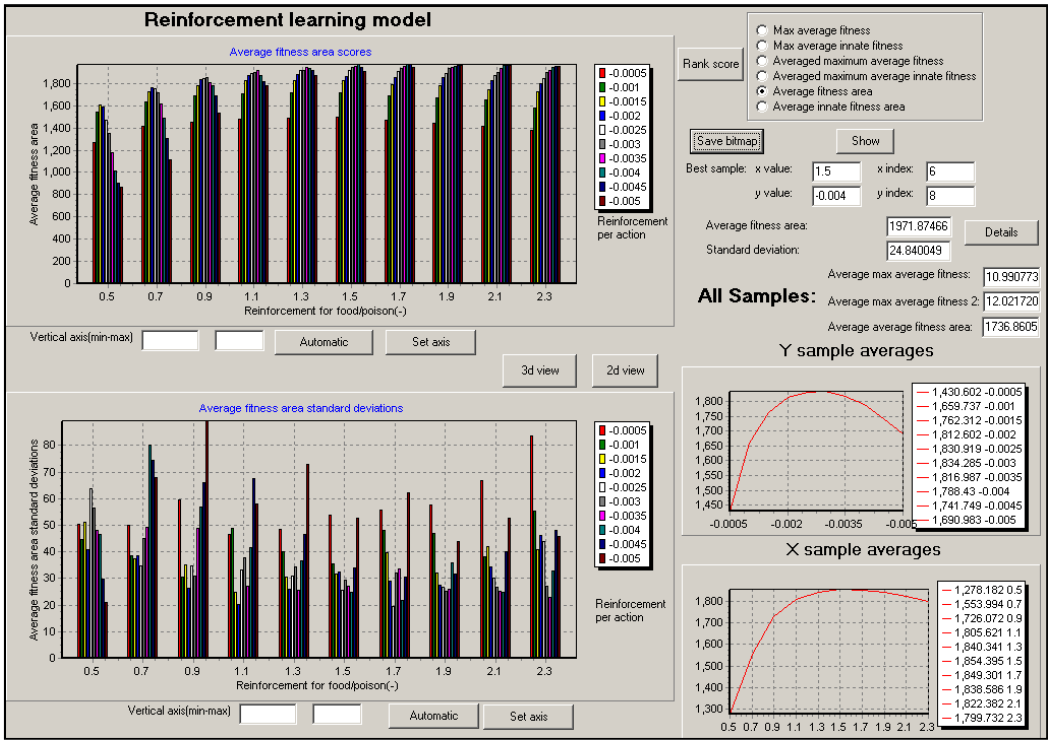


Figure 6-8 Performance results using feedback values in the range [0.5…2.3] for locating food and [-0.5…-2.3] for locating poison, with reinforcements per action being investigated in the range [-0.0005…-0.005].

Our empirical investigation used feedback values in the range [0.5 … 2.3] for locating food and [-0.5 … -2.3] for locating poison, with increments of +0.2 and –0.2 in either case. Feedback per action was investigated in the range [-0.0005 … -0.005] with increments of –0.0005. The results of the investigation led us to use positive reinforcements of 1.5 and negative reinforcements of –1.5 for food and poison events, with a small constant, negative reinforcement of –0.003 being used per action. The chosen values are mostly influenced by the important average fitness area data (Figure 6-8 above), note the x and y axis sample averages, with the most successful parameter values produced by the other measures differing only slightly from these. Refer to appendix E for complete data.

The following presents the evaluation of our reinforcement learning model in an environment that ranges from stable to highly unstable.

6.3 Evaluation of Our Reinforcement Learning Model

6.3.1 Performance in Stable to Unstable Environments

6.3.1.1 Overview

The following presents our reinforcement learning model's performance in response to differing levels of environmental stability, ranging from stable to highly unstable.

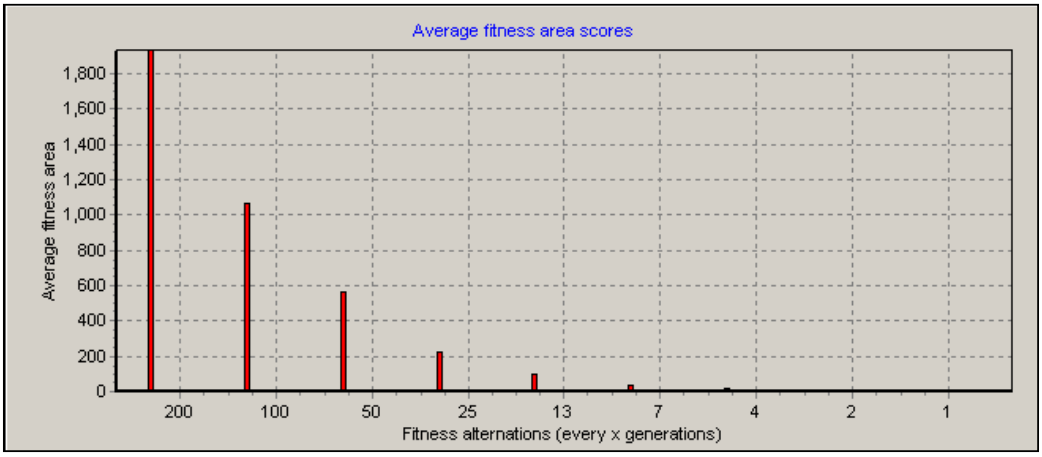


Figure 6-9 Average fitness area scores produced by our reinforcement learning model in stable to highly unstable environments.

As is shown by Figure 6-9, increasing levels of instability result in a corresponding decrease in the model's ability to adapt. This performance trend is similar to that produced by our evolution model and, thus, indicates that it too is ineffective in dealing with increasing levels of instability. The above data concerns the important average fitness area scores, considered to be the most informative in terms of consistency of performance. Refer to appendix F for complete data.

To investigate the adaptive nature of the model further, the following presents details regarding its performance in response to several different degrees of environmental stability.

6.3.1.2 Performance Details

6.3.1.2.1 Stable environment

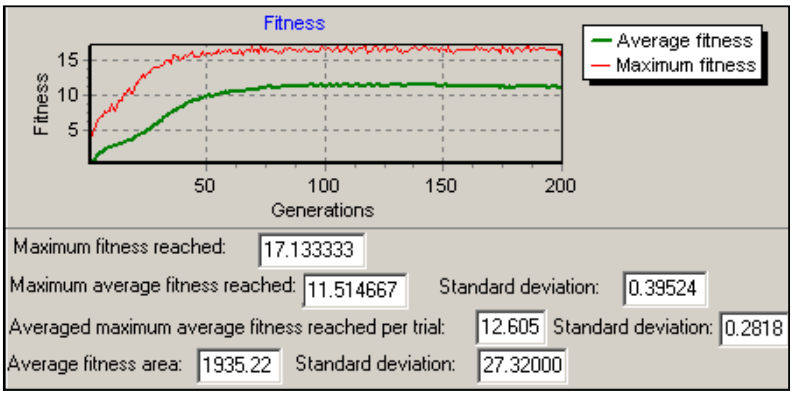


Figure 6-10 The reinforcement learning model's performance in the stable environment.

The reinforcement learning model is clearly effective in a stable environment (Figure 6-10), reaching peak fitness levels within 100 generations. Recall that the evolution model's performance was also effective, however, it demonstrated a steady progression towards maximum fitness levels over the full 200 generations (refer to chapter 10 for a more in depth comparison). Our reinforcement learning model, on the other hand, demonstrates that it can reach its maximum effectiveness more quickly, in addition to producing higher fitness levels. The superior performance is due to lifetime learning's potential to adapt at a greater speed, constantly utilizing learning to modify an agent's behaviour within their lifetime. It seems that the stable environment allows this potential to be realized. However, how does our model's individual reinforcement learning ability respond to increasing levels of instability?

6.3.1.2.2 Fitness Consequence Alternations Every 25 Generations

Figure 6-11 The reinforcement learning model's performance in an environment with fitness consequence alternations every 25 generations.

The above figure (Figure 6-11) represents our reinforcement learning model's performance in an environment that undergoes fitness consequence reversals every 25 generations. Unlike the evolution model, our reinforcement learning model demonstrates an ability to quickly unlearn incorrect behaviour (i.e., locating poison objects that were previously food objects), due to negative reinforcements. This has the effect of reducing the impact of the fitness consequence reversals. Recall that the evolution model took a number of generations to unlearn the incorrect behaviour, producing negative fitness consequences during this time.

However, although the reinforcement learning model is effective in quickly unlearning incorrect behaviour, the agents are still required to relearn appropriate behaviour, previously incorrect and unlearnt. Although faster than the evolution model, this is a time consuming process that does not improve as the generations progress.

Thus, the model is limited in its ability to deal with such changes, as it cannot remember conflicting solutions across the generations. Each time a fitness consequence occurs the agents are required to relearn their environment with no reduction in learning time. As a result, no assimilation of the changes occurs and, therefore, there is no ultimate growth in the maximum fitness levels.

These characteristics are emphasized by further increases in instability.

6.3.1.2.3 Fitness Consequence Alternations Every 7 Generations

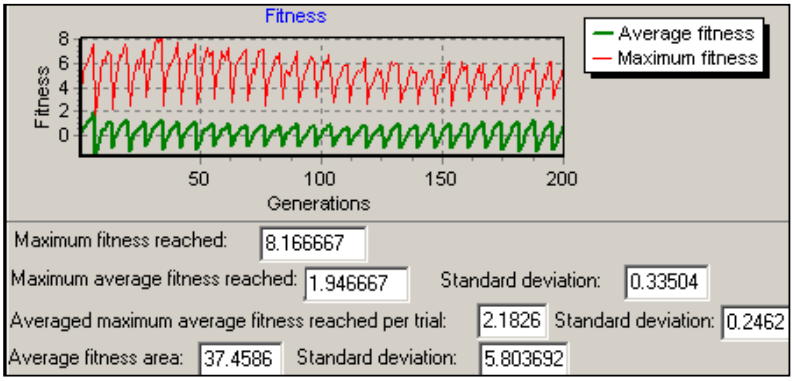


Figure 6-12 The reinforcement learning model's performance in an environment with fitness consequence alternations every 7 generations.

Due to the nature of the reinforcement learning model's limited ability to respond to changes, increasing the frequency of the fitness consequence reversals results in a highly unstable, oscillating performance (Figure 6-12). Note that throughout the duration of the simulation there is no improvement in the fitness peaks, in terms of both average and maximum fitness levels. However, this performance is slightly better than that of the evolution model, due to its more effective unlearning and relearning ability, resulting in a positive average fitness area. Thus, individual learning of this type does respond to the changes faster than that of the evolution model, but it is ineffective in its ability to assimilate such changes in order to perform more effectively over the course of the simulation. The changes are disruptive, and further exposure to them does not result in any improvements.

6.3.1.2.4 Fitness Consequence Alternations After Every Generation

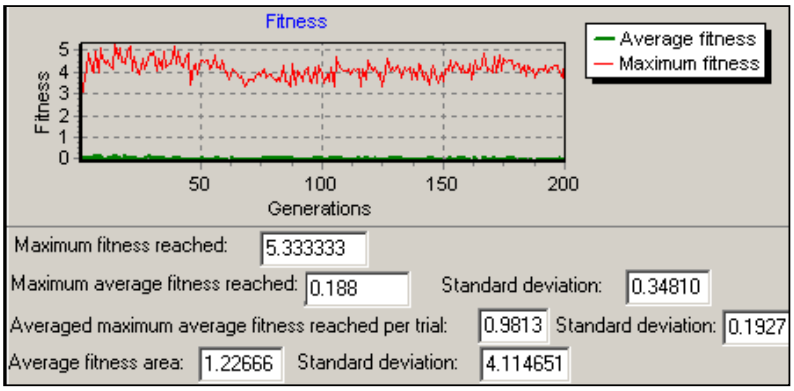


Figure 6-13 The reinforcement learning model's performance in an environment with fitness consequence alternations after every generation.

In response to the highest level of instability, with fitness consequence reversals occurring after every generation, the reinforcement learning model is incapable of

learning and produces average fitness scores only slightly greater than 0 throughout the simulation (Figure 6-13).

The nature of this performance differs from that of the evolution model's oscillating behaviour, demonstrating increases in amplitude as the simulation progressed and ultimately producing a negative average fitness area. In this case, the reinforcement learning model's adaptive response is quite different. Slightly positive average fitness levels are consistently maintained throughout the simulation. However, no growth occurs in either these or the agents' maximum fitness levels.

Thus, it is also unsuccessful in adapting to such situations, with virtually no learning occurring and ultimately producing an average fitness area that is only slightly positive.

The following presents an investigation of the adaptive behaviour and knowledge of the agents produced by our reinforcement learning model in order to help us better understand the above results and the model's adaptive nature.

6.3.2 Adaptive Behaviour and Neural Network Knowledge

Our reinforcement learning model produces agents that possess the ability to modify their behaviour according to their experience of the environment. This is achieved via the application of positive and negative reinforcements in relation to the desired objectives: locate food and avoid poison. In response to a stable environment the model quickly produces agents that contain the necessary neural network knowledge to behave appropriately. However, under increasing levels of instability the model's performances rapidly degrades to ultimately be considered ineffective.

The following is an investigation of the behaviour and neural network knowledge produced by our reinforcement learning model in order to provide a more complete explanation of its performance and further to emphasize the advantages and limitations mentioned in the previous section.

We begin with a trace of an effective agent produced in an environment demonstrating stable characteristics.

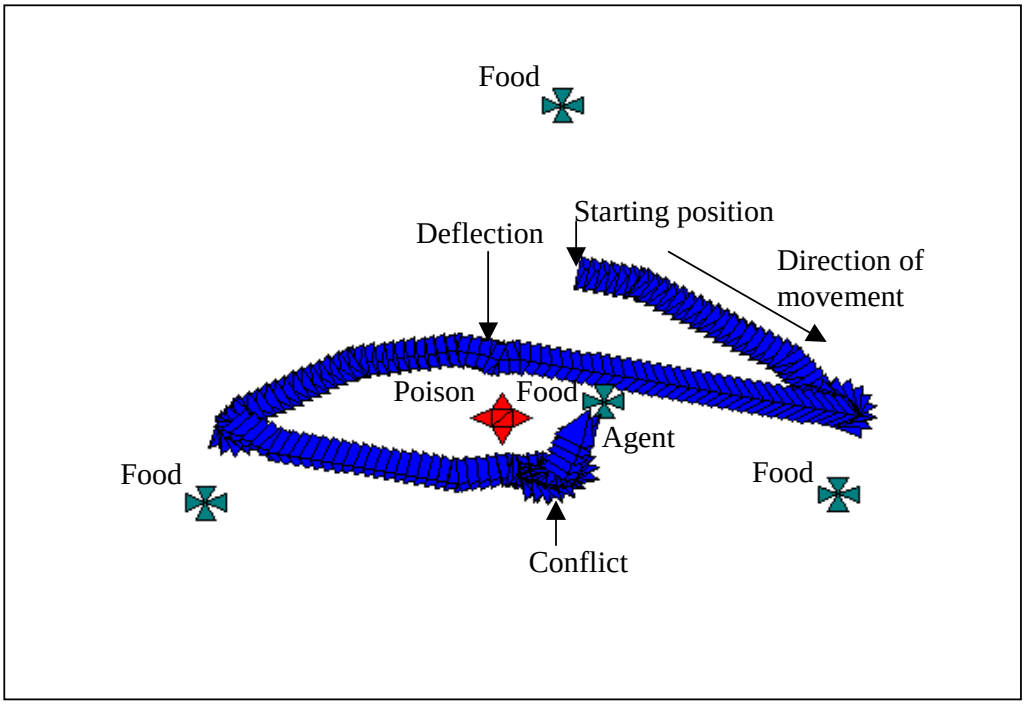


Figure 6-14 A trace of a successfully reinforced agent demonstrating its ability to locate food and a tendency to avoid poison – note the deflection and conflict.

The trace of Figure 6-14 was taken of an agent after 195 generations and demonstrates its ability to locate food and a tendency to avoid poison, as indicated by the deflection in the agent's path caused by the proximity of poison and the conflict generated by the agent's resistance to turn towards poison in order to obtain the food object. Note that the conflict resolution ability is a particular feature of the agent's individual

reinforcement learning that cannot be produced by the evolution model. It is a product of the small, constant negative reinforcements that force the agent to attempt a different action should its behaviour result in a conflict between actions (i.e., repeated left and right rotations). It provides a mechanism of weakening such mappings to produce a different action and, thereby, resolve the conflict.

For example, consider the situation represented in the above diagram. The agent finds itself in a position where a leftward rotation towards food places itself in a position that maps to a rightward rotation away from the poison object. This, in turn, results in another leftward rotation and the cycle continues. The evolution model does not have much defence against such vicious circles, except the fact that the neural network mappings are not deterministic and, therefore, provide the possibility of another action taking place, even if it only has a small probability of occurring. However, the small, constant negative reinforcements provide a more effective mechanism. Should an agent be caught in such a position, then these reinforcements will weaken the mappings that are involved in producing the conflicting behaviour. As a result, a different situation-action mapping can be made more probable and the situation resolved. This, then, is an aspect of the reinforcement learning model that can potentially produce more effective solutions than those of the evolution model.

In addition, the reinforcement learning model can produce superior performances, since its feedback is more informative in directing the agents towards effective behaviour. Recall that the evolution model only makes use of fitness scores at the end of each generation, to produce the following population. Our reinforcement learning model utilizes the negative and positive reinforcements, in respect to locating poison and food, to modify the agent's behaviour as soon as these events occur and in relation to those neural network connections that are the most responsible for these events occurring. The evolution model can only judge the individual agents as a whole, according to their fitness. Thus, the reinforcement learning model can demonstrate a superior learning ability, resulting in the more effective performance.

The following provides a more specific investigation of an agent's knowledge by examining its neural network connection strengths. Note that although the nature of the reinforcement learning model differs in certain respects to that of the evolution model they are both confined to searching the same neural network weight space. The models differ in terms of the speed at which they can modify the neural networks and the manner in which they do so.

The following two diagrams represent a successfully reinforced agent's neural network knowledge after 198 generations in the stable environment.

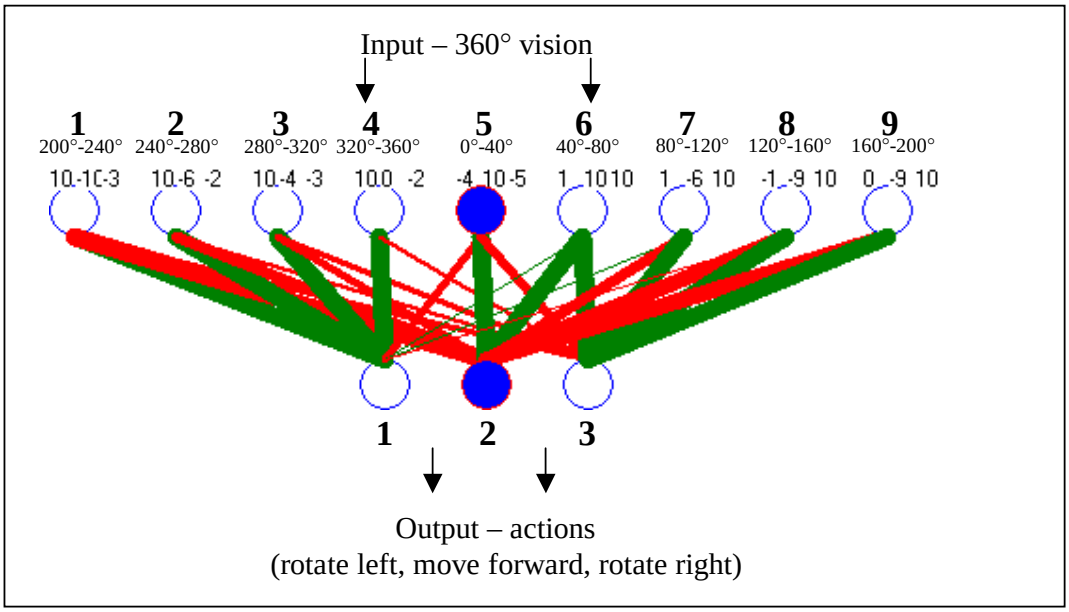


Figure 6-15 A successfully reinforced neural network responsible for mapping visual input regarding green objects (food in the stable environment) to an output action.

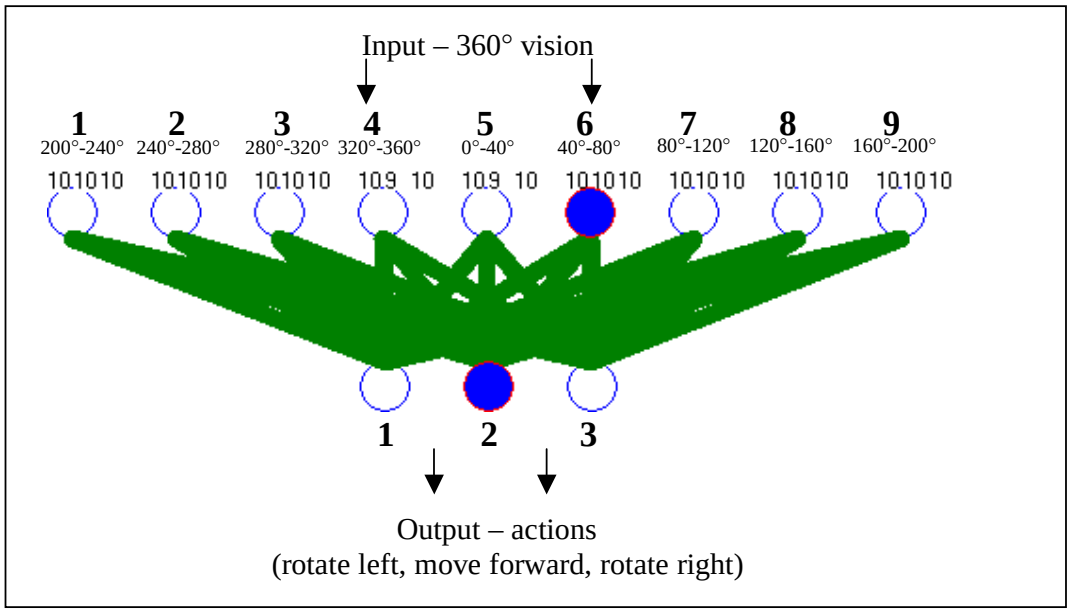


Figure 6-16 A successfully reinforced neural network responsible for mapping visual input regarding red objects (poison in the stable environment) to an output action.

Figures 6-15 and 6-16 represent the portions of a successfully reinforced agent's neural network responsible for mapping visual input regarding green and red objects (food and poison, respectively, in the stable environment) to an output action.

The following figures were produced from the above neural network information and represent the agent's actions that are most likely to occur given the position of the objects relative to the agent.

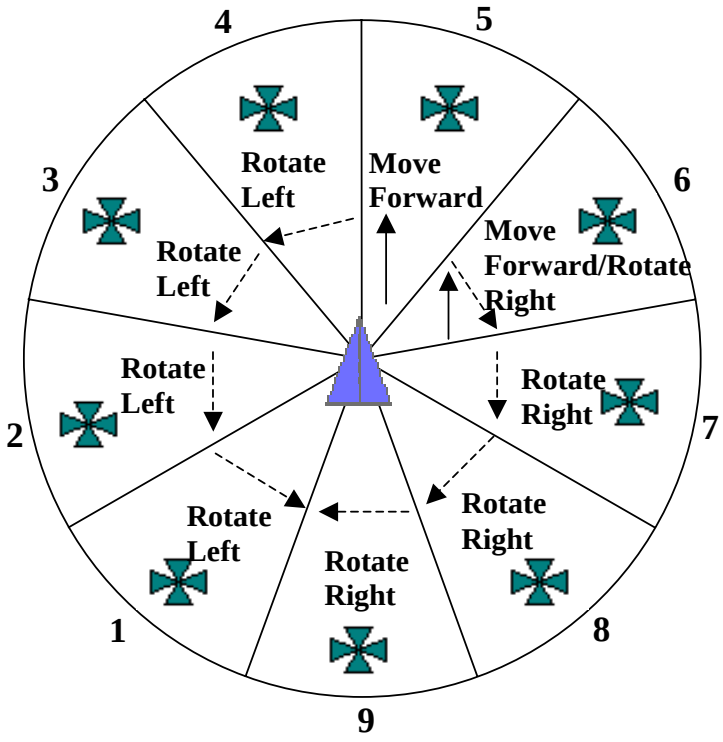


Figure 6-17 The influence of the position of green objects (food in the stable environment) on the agent's behaviour.

The figure above (Figure 6-17) reveals how the agent navigates itself towards green objects (food). Recall that regions 4 and 5 are responsible for the agent's frontal vision (between 320°and 40°). The agent demonstrates a tendency to prefer the object in region 5 before moving forward with the regions to its left (1,2,3 and 4) producing leftward rotations and the regions to its right (6,7,8, and 9) mostly producing rightward rotations, the exception being region 6 that has an equal probability of mapping to forward movements and rightward rotations. Thus, the solution is similar to that produced by the evolution model. However, a notable difference is that the neural network connection strengths of Figure 6-15 are much stronger, due to the more constant learning.

The portion of the agent's neural network responsible for mapping red objects to an action is perhaps more unusual in comparison to that of the evolution model. Using the information provided in Figure 6-16 we can produce the following diagram (Figure 6-18).

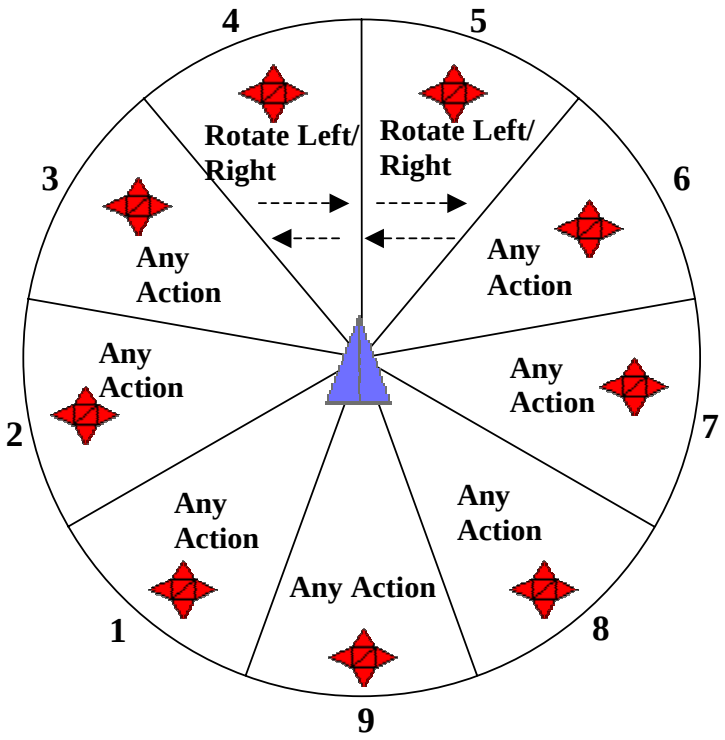


Figure 6-18 The influence of the position of red objects (poison in the stable environment) on the agent's behaviour.

In all regions besides 4 and 5 the agent demonstrates no preference towards any particular action. This can be explained as being due to the nature of the reinforcement learning model to positively reinforce all previous mappings occurring prior to the location of food, including the connections between input neurons responsible for seeing poison and the chosen output action. Since poison can occur in any particular position relative to the agent when it locates food, the connections between the neurons responsible for seeing poison and all previous chosen actions responsible for locating food, will be positively reinforced. As a result, maximum connection strengths will eventually be reached and each action will be equally influenced by the position of poison. However, should poison be eaten, perhaps necessary to locate food, then those mappings responsible will be negatively reinforced, i.e., the mappings from region 4 and 5 above to move forward actions, resulting in a greater tendency to avoid poison in such positions by rather making leftward or rightward rotations more probable. This can explain the deflection and conflict contained in the trace of Figure 6-14.

The model, then, is clearly effective in producing successful neural networks in the environment demonstrating stable characteristics. However, the situation changes in response to instability. The following relates our previous statements regarding the model's limited ability in learning and applying conflicting solutions and its unlearning and relearning times in relation to the agent's neural networks.

Due to negative reinforcements, the model demonstrates an effective ability to unlearn previous behaviour. The strong neural network connections of Figure 6-15 can be quickly unlearnt, since their involvement in producing behaviour that becomes incorrect, due to a fitness consequence reversal, can be directly targeted. Thus, in this sense the model is superior to our evolution model. For the same reason, it also demonstrates a superior ability to relearn the correct behaviour. However, such unlearning and relearning demonstrates no ability to assimilate the changes and improve learning times as the simulation progresses. Thus, this model, like the evolution model, is limited in its ability to learn and apply solutions of a conflicting nature.

6.4 Conclusion

Our reinforcement learning model demonstrates an effective ability, superior to that of the evolution model, to produce successful agents in response to an environment demonstrating stable circumstances. Although the feedback is of a similar, limited nature as that given to the evolution model, in this case learning takes place during an individual's lifetime, as opposed to the generation timescale of our evolution model. In addition, the feedback is utilized to target particular aspects of the agent's neural network. In comparison, our evolution model utilized fitness scores to judge agents as a whole in order to produce a new generation. Thus, the reinforcement learning model is capable of producing results of a greater effectiveness than the evolution model over the same number of generations.

However, its performance in response to increasingly unstable environments ultimately degrades to become as ineffective as that of the evolution model. Although learning in this case is applied with a greater frequency, it cannot help the model deal any more effectively with the dynamics of the most unstable environments. The reinforcement and evolution model both face the problem of learning a relationship between an object and its fitness or reinforcement value that needs to be unlearnt in a later generation, in the extreme case, after every generation. Although the reinforcement learning model does demonstrate superior unlearning and relearning times, it demonstrates no ability to assimilate the changes and improve learning times as the simulation progresses. Thus, like the evolution model it is limited in its ability to learn and apply solutions of a conflicting nature.

Section 3 - Design and Evaluation of Evolution and Reinforcement Learning Models

As the results of the previous two chapters indicate, our evolution and reinforcement learning models are quite effective in response to stable conditions. However, under conditions of increasing instability their effectiveness is limited. The models were found to be incapable of learning and applying solutions of a conflicting nature. Our investigation led us to conclude that this is due to an insufficient unlearning/relearning ability that demonstrates no assimilation of changes.

This section presents our work investigating hybrid evolution and reinforcement learning models, in Darwinian and Lamarckian frameworks, with a view to extracting their advantages and disadvantages in dealing with an environment demonstrating stable to highly unstable characteristics.

The following three chapters present our:

- Design of hybrid evolution and reinforcement learning models, in Darwinian, Lamarckian and combined Darwinian/Lamarckian frameworks, with a view towards maximizing their ability to produce adaptive agents in an initially stable to unstable environment.
- Evaluation of the effectiveness and adaptive nature of the models with regard to their performance in stable conditions and in terms of their response to increasing levels of instability.

Chapter 7: Design and Evaluation of a Darwinian Evolution and Reinforcement Learning Model

7.1 Introduction

Under a Darwinian framework, what is learnt during an individual's life is not inherited by their offspring. As a result, knowledge acquisition occurs at two levels, the population or generation level and the individual level. The former utilizes the processes of evolution to alter knowledge encoded in the individual's genotype, while the latter uses an individual's ability to learn during its lifetime and according to its experience of the environment. Thus, combining evolution and learning in a Darwinian framework requires that evolved (genotype) knowledge is preserved and the individual's learnt (phenotype) knowledge is not inherited by their offspring.

This separation does have its advantages. The population's global search is not directly influenced by the individuals' local search. As a result, the benefits of both global and local search processes can be effectively maintained. The population can sample large areas of the solution space with the individuals' learning ability providing a more effective search over smaller areas.

In addition, since evolution works over a larger time scale, maintaining genetic knowledge can be beneficial in preserving relevant, long-term information regarding the environment in contrast to an individual's preoccupation with its limited experience of the environment. Thus, maintaining knowledge produced by the population's experience over many generations can prove useful, since such knowledge can then be refined by the individuals' learning ability to effectively adapt to its particular experience of the environment. Thus, a Darwinian model can utilize evolution to extract a range of useful, general behaviour that can be further refined by reinforcement learning to meet the particular adaptive requirements of the environment an agent may be born into.

Lastly, the Baldwin Effect can be an advantageous characteristic of Darwinian evolution in terms of learning's ability to indirectly guide evolution towards effective solutions. Individuals who are more predisposed to learning useful knowledge will achieve greater fitness levels than those who are not.

An additional, general advantage of both Darwinian and Lamarckian evolution and learning models is the ability of the evolution process to adjust parameters of the learning algorithm according to the requirements of the environment by including such parameters in the individuals' genotype descriptions.

Our investigation examines whether such adaptive characteristics can help our Darwinian evolution and reinforcement learning model deal more effectively with the environment than the previous models.

The following presents our:

- The design of a Darwinian evolution and reinforcement learning model with a view towards maximizing its ability to produce adaptive agents in an initially stable environment that becomes increasingly unstable.
- An evaluation of the effectiveness and adaptive nature of the model with a view towards extracting the advantages and limitations of such models in dealing with stable environments that become increasingly unstable.

7.2 Design

The following is required of our model:

- A synthesis of the evolution and reinforcement learning processes described in the previous chapters.
- A separation of knowledge at the genotype and phenotype level in order to accomplish Darwinian evolution.
- The ability to adjust, via evolution, the feedback parameters utilized by our reinforcement learning algorithm.

7.2.1 Synthesis

A synthesis is quite easily achieved by applying our reinforcement learning algorithm throughout the agents' lifetime and then generating a new population, based on their fitness scores, by using our evolution algorithm's processes of selection and mutation. Feedback is still of the same form, however now the agents receive reinforcements throughout their lifetime and accrue fitness scores in relation to the same events. Thus, the agent knowledge is modified as a result of both evolution and learning.

7.2.2 Genotype/Phenotype Separation of Knowledge

Since we wish to create a Darwinian model, it is required that individually learnt knowledge is not inherited by offspring and evolved (genotype) knowledge is preserved across generations, to be modified only by the mutation operator. Thus, an agent's knowledge must be separated at a genotype and phenotype level.

In order to achieve this, our agents possess an innate neural network description (the genotype) containing weight values. At the start of each generation the agents begin their life possessing this knowledge. All changes to the agent's neural network weights, as a result of reinforcement learning, are not applied to the genotype. Rather, a temporary, learnable weight value is used to accomplish learning at the phenotype level.

At the start of the agent's life the learnable weight values are set equal to the inherited weight values. All neural network mappings and reinforcement learning operations utilize these values. At the end of an agent's life all learnt changes are lost and offspring are produced using the innate, genetic weight values of their parents, although these may be altered by small mutations to modify the evolving, genotype knowledge. Thus, a distinction is accomplished between learnt knowledge and evolved knowledge (phenotype and genotype).

Note though, that it is the phenotype that is selected by the evolution process, according to its fitness, and the genotype that is reproduced. Therefore, genotypes replicate in proportion to their phenotype's effectiveness. As a result, genotypes are selected in relation to their ability to produce effective phenotypes (i.e., complement learning) and it is this interaction, then, that can produce successful, complementary relationships between evolution and learning.

Our evolution process can also complement reinforcement learning by adjusting its feedback parameters.

7.2.3 Adjusting Reinforcement Learning's Feedback Parameters

Feedback parameters can be made a part of the evolution process by including them in an agent's genotype description. By doing so they can be adjusted like any other attribute defining an agent, via the processes of selection and mutation, according to the degree of learning required. This provides an advantage over fixed values, since the stage of the simulation or changes in the environment's stability may require different feedback values. Thus, such optimization may prove useful in dealing with our environments of increasing instability.

Should an agent be selected for mutation then the change in our agent's feedback parameters can be determined according to:

$$\Delta\text{feedback parameter} = \begin{cases} x & \text{if a random digit is 0} \\ 0 & \text{if the random digit is 1} \\ -x & \text{if the random digit is 2} \end{cases}$$

where x is a small, constant value.

This can be applied to our 3 feedback parameters, reinforcement for locating green objects, reinforcement for locating red objects and reinforcement per action.

In order to find an effective degree of change (x), experiments were performed using initial feedback values of 0 and utilizing degrees of change in the range 0.05 to 0.5 for food/poison reinforcements (green/red objects in the stable environment) and in the range 0.001 to 0.01 for feedback per action adjustments. Results are contained in appendix G. Based on this data we utilize movements of ±0.2 for food/poison feedback and ±0.003 for feedback per action.

7.2.4 Simulation Framework and Parameters

Our Darwinian evolution and reinforcement learning model is applied using the same simulation framework as that of our previous models.

The important constants are:

- A generation lifetime of 500
- A simulation length of 200 generations
- A population of 25
- 1 food object in the world at any time
- 1 poison object in the world at any time
- 30 trials per sample

Due to the increased complexity of this model, several additional options are now available. We have the option of using fixed reinforcement values with a fixed individual mutation rate or a mutation rate that is allowed to self-optimize or we can allow the reinforcement values to be adjusted in conjunction with a fixed mutation rate or a mutation rate that is allowed to self-optimize. These can prove to be useful options with regard to the model's adaptive ability, with particular relevance to changes in stability.

However, in order to choose appropriate fixed reinforcement learning feedback parameters we have the option of using the fixed population and individual mutation rates of 0.8 and 0.1, previously shown to be successful, or we can allow the individual mutation rate to self-optimize. Refer to appendices H and I for complete data regarding these experiments. The results from both sets of data are almost equivalent, however, utilizing a fixed individual mutation rate does produce slightly superior measures and, therefore, this data is used to choose our fixed reinforcement values of 0.4, -0.4 and –0.02 for locating food, poison and per action, respectively.

These then are the fixed reinforcements that are used, in conjunction with a fixed mutation rate and a mutation rate that is allowed to self-optimize, in order to evaluate our Darwinian evolution and reinforcement learning model's response to an initially stable environment that becomes increasingly unstable. Experiments that adjust the reinforcements in conjunction with a fixed mutation rate and a mutation rate that is allowed to self-optimize were also performed. Refer to appendix J for complete data regarding these experiments.

Note that, in the environments utilizing fitness consequence alternations, the sign of the reinforcement values are also reversed, maintaining consistency with the fitness consequence reversals. Thus, our Darwinian evolution and reinforcement learning model is provided with appropriate reinforcement and fitness feedback in order to train our agents to maximize their ability to locate food and avoid poison.

7.3 Evaluation of Our Darwinian Evolution and Reinforcement Learning Model

7.3.1 Performance in Stable to Unstable Environments

7.3.1.1 Overview

The following presents our Darwinian evolution and reinforcement learning model's performance in response to differing levels of environmental stability, ranging from stable to highly unstable.

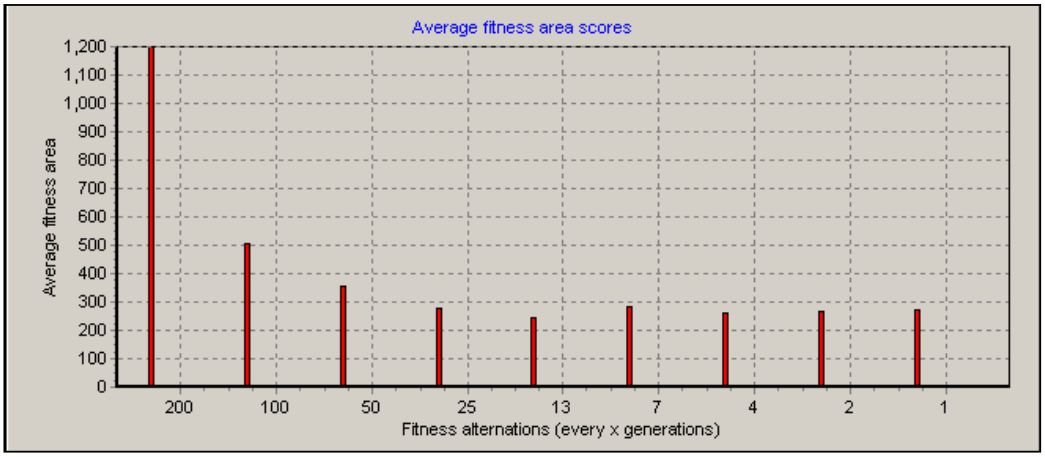


Figure 7-1 Average fitness area scores produced by our Darwinian evolution and reinforcement learning model in stable to highly unstable environments.

The most successful performance (Figure 7-1) was produced using a fixed mutation rate and adjusting reinforcement values, starting with initial values of 0.4, -0.4 and –0.02 for locating green objects (food in the stable environment), red objects (poison in the stable environment) and per action, respectively, with a population mutation rate of 0.8 and a fixed individual mutation rate of 0.1. Refer to appendix J for complete data concerning the other options (optimising reinforcements and the individual's mutation rate, using fixed reinforcements and a fixed individual mutation rate and, lastly, using fixed reinforcements and allowing the individual's mutation rate to self-optimize). With reference to the other options, it is important to note that fixed reinforcements appear to be slightly more effective in the most stable environments. However, they are ineffective in dealing with increasing levels of instability, producing a performance similar to that of the reinforcement learning model. Allowing the reinforcement feedback values to be optimized, though, provides a far more effective performance in response to increasing levels of instability.

As before, average fitness area scores are presented, the most informative in terms of consistency of performance. Complete data regarding the other performance measures is contained in appendix J.

As the above performance indicates, our Darwinian evolution and reinforcement learning model does not decay into ineffectiveness as a result of increasing instability.

Rather, it appears to reach a consistent level of effectiveness in response to increasing levels of instability. Note that the effectiveness levels of the model in an environment utilizing fitness consequence reversals every 25 generations is maintained under increasing levels of instability, even in response to the most unstable conditions. It appears that the model's increasing exposure to instability results in an improved ability to assimilate the environment's changes. In comparison, the evolution model and the reinforcement learning model were both incapable of producing any level of effectiveness beyond fitness consequence reversals every 13 generations. Recall the negative average fitness area scores of the evolution model and the completely ineffective scores of the reinforcement learning model (i.e., near 0). A more direct comparison is provided in chapter 10.

Thus, our Darwinian evolution and reinforcement learning model demonstrates an ability to assimilate the increasing instability and, thereby maintaining a consistent performance even under increasing levels of change.

The following provides performance details regarding the model's performance in response to several degrees of environmental instability.

7.3.1.2 Performance Details

7.3.1.2.1 Stable Environment

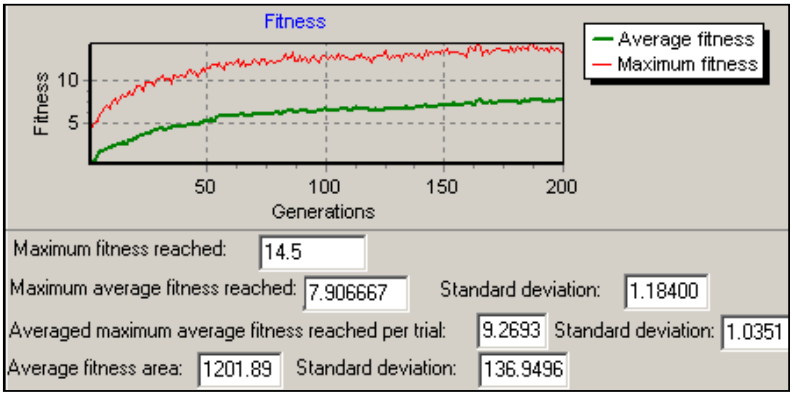


Figure 7-2 The Darwinian evolution and reinforcement learning model's performance in the stable environment.

Our Darwinian evolution and reinforcement model performs adequately in the stable environment (Figure 7-2). However, it is not more effective than either the evolution model or the reinforcement learning model under such conditions. In fact, even utilizing the most effective fixed reinforcements, which produced the best sample for this model's performance in the stable environment, its effectiveness could only be considered slightly better than that of the evolution model. Thus, learning does not appear to help the Darwinian evolution process in the stable environment, especially if the reinforcement feedback values are allowed to adjust.

The following figure (Figure 7-3) includes innate fitness measures, produced from evaluations of the agents' innate, genotype knowledge prior to learning. To achieve this the population is examined twice. The agents are first evaluated utilizing only their genotype knowledge to produce behaviour. Selection is not applied and the same population re-evaluated with individual reinforcement learning. A new population is then produced using the fitness scores of the learning agents and the process repeated. This allows us to determine the effect that learning is having on the evolution process, as compared to just seeing the combined effectiveness of both processes.

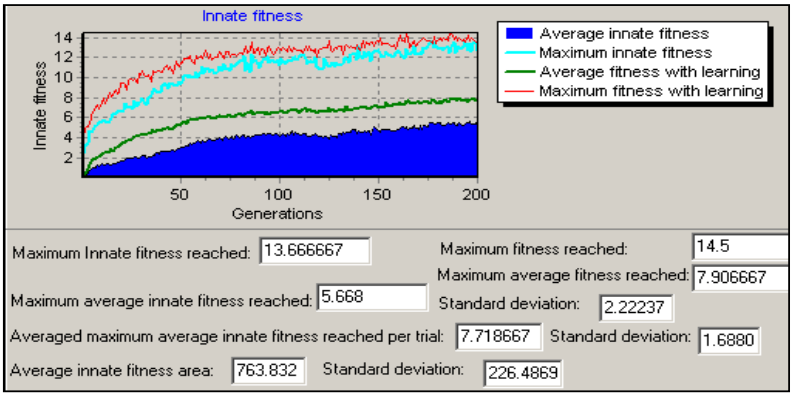


Figure 7-3 The Darwinian evolution and reinforcement learning model's innate and learning performance in the stable environment.

With reference to the above figure (Figure 7-3), we draw your attention to the agents' average innate fitness graph (solid, blue). This is an evaluation of the agents' behaviour as produced by their innate, genotype knowledge. The average fitness of the agents, utilizing both their genotype and learnt phenotype knowledge, is also represented (green). Note that leaning appears to provide the agents with an adaptive advantage. Their fixed, evolved knowledge is clearly less effective without the benefit of learning. However, with reference to the fixed knowledge produced by our evolution model, both the evolved knowledge and the combined innate and learnt knowledge, presented above, are not superior to that of the evolution model's performance. Thus, there is no sign of the Baldwin effect guiding the agents' innate knowledge, through learning, to produce results more effective than those of evolution alone.

Rather, the above results could be due to innate knowledge being evolved to rely on learning taking place. Should learning be removed then the innate knowledge will produce solutions that are naturally less useful and, therefore, less effective. In this sense learning has moved the genetic knowledge to regions of the solution space that results in its effectiveness being reduced should learning not occur. This is related to our previous description regarding the nature of learning to potentially hinder the evolution processes search by hiding the effectiveness of innate, genotype knowledge (as described in section 3.5).
Thus, learning does not appear to be of much use to our Darwinian evolution model in adapting to the stable environment.

However, it does appear to be highly useful under conditions of instability.

7.3.1.2.2 Fitness Consequence Alternations Every 25 Generations

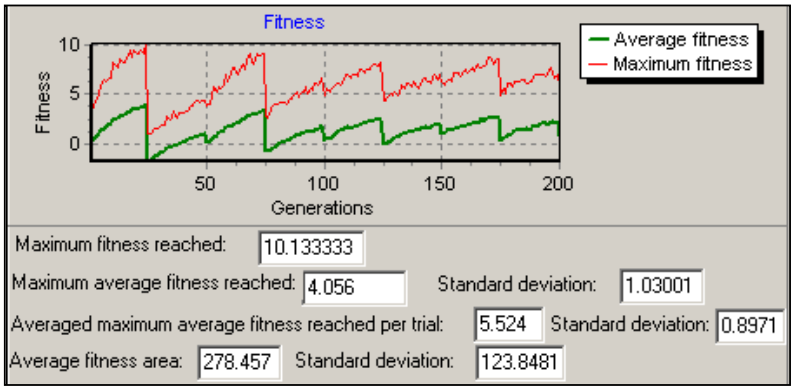


Figure 7-4 The Darwinian evolution and reinforcement learning model's performance in an environment with fitness consequence alternations every 25 generations.

The characteristics of our Darwinian evolution and reinforcement learning model's response to fitness consequence alternations every 25 generations (Figure 7-4) reveals certain advantages over the previous models.

Although the fitness consequence reversals are disruptive, the agents now demonstrate a greater ability to maintain positive average fitness levels. However, peak fitness levels do not improve over the course of generations. It appears that at this level of instability the agents have not yet had sufficient experience of the changes in order to improve learning times over the duration of the simulation, although an analysis of the agents' genotype knowledge does reveal that a certain amount of adaptation is occurring.

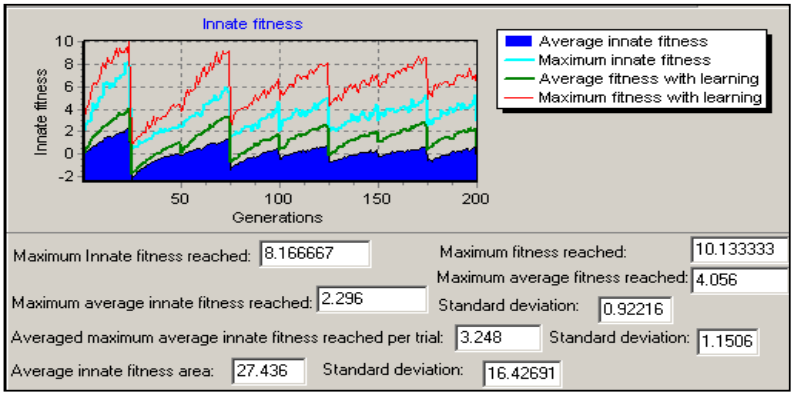


Figure 7-5 The Darwinian evolution and reinforcement learning model's innate and learning performance in an environment with fitness consequence alternations every 25 generations.

The average innate fitness levels of Figure 7-5 indicates that the agents' genotype knowledge is initially following that which is being learnt, i.e., producing knowledge to locate either green or red objects and suffering equivalently after a fitness consequence reversal – producing negative average fitness scores. However, after several fitness consequence reversals have occurred, our agents' innate, genotype

knowledge effectiveness, prior to learning, is more inclined to maintain a performance evaluation near 0, relying on learning to produce effective fitness levels. It is this that allows our agents to maintain positive average fitness levels, since under a Darwinian framework offspring lose the learnt knowledge of their parents, i.e., incorrect solutions after a fitness consequence reversal, starting their life utilizing genotype knowledge with an effectiveness near 0. Both our evolution model and reinforcement learning model maintain the knowledge of previous generations and are, thus, slow to unlearn incorrect solutions, resulting in negative average fitness scores.

The Darwinian evolution and reinforcement learning model can, then, provide a mechanism where incorrect solutions are not inherited and genotype knowledge can be evolved to complement the effectiveness of learning. In the above case, producing consistently positive average fitness levels even after the occurrence of fitness consequence reversals.

However, it is not merely the case of the agents' genotype effectiveness without learning being maintained near 0 in order to buffer against incorrect solutions after fitness consequence reversals. The evolved knowledge may not be following learning, but that does not mean that it is standing still. The following reveals how genotype knowledge, after exposure to a greater frequency of fitness consequence reversals, can be evolved to improve learning times in response to increasing instability. It does this not by following learning, but by complementing it.

7.3.1.2.3 Fitness Consequence Alternations Every 7 Generations

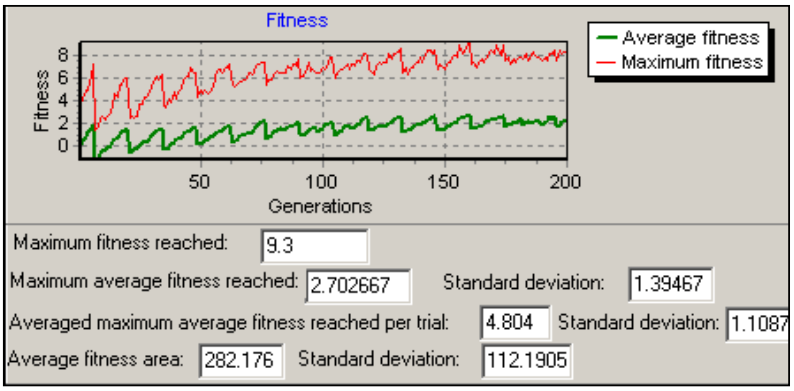


Figure 7-6 The Darwinian evolution and reinforcement learning model's performance in an environment with fitness consequence alternations every 7 generations.

Figure 7-6 demonstrates that the Darwinian evolution and reinforcement learning model has the ability to produce agents that can effectively assimilate fitness consequence reversals to produce an improving performance over the entire course of the simulation. It appears that knowledge is being produced to help the agents learn more effectively in response to the environment's changes, resulting in a more stable and improving performance.

The following figure (Figure 7-7) includes an evaluation of the agents' innate knowledge, allowing us to investigate the role that it is playing in the improved effectiveness.

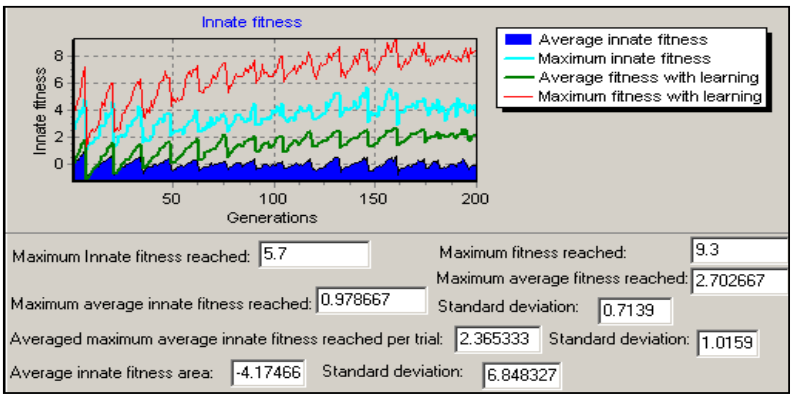


Figure 7-7 The Darwinian evolution and reinforcement learning model's innate and learning performance in an environment with fitness consequence alternations every 7 generations.

Note that the evaluation of the agents' innate, genotype knowledge (blue, solid graph) initially appears to follow the performance of the agents utilizing learning and their evolved knowledge (green graph), oscillating to an equivalent level. That is, knowledge is being evolved to locate the appropriate object of that particular period and, thus, suffering from the fitness consequence reversals. However, after a number of generations, the nature of the performance begins to differ.

The agents' effectiveness begins to steadily increase while its innate knowledge maintains an evaluation near 0. It appears, then, that the agents are being evolved to contain innate knowledge that is not biased towards either of the two solutions, locate green objects and avoid red objects or locate red objects and avoid green objects. However, this does not mean that useful knowledge is not being evolved. Note that the agents' average fitness levels improve the course of generations while their genotype knowledge, prior to learning, maintains an effectiveness near 0. Recall that these improvements cannot be due to learning alone, since under a Darwinian framework all learnt knowledge is lost at the end of a generation. It must, then, be the knowledge contained in the agents' genotype that is leading to these improvements. Thus, in contrast to the evolved knowledge following learning it is, rather, of a different nature and complements it.

In the following behaviour and neural network knowledge section we will demonstrate that the nature of this knowledge is to encode general behaviour to locate either object. The above performance is possible as the agents can utilize such knowledge, in conjunction with their individual learning ability, to adapt to their particular situation.

This is true of even the most unstable environment, utilizing fitness consequence reversals after every generation.

7.3.1.2.4 Fitness Consequence Alternations After Every Generation

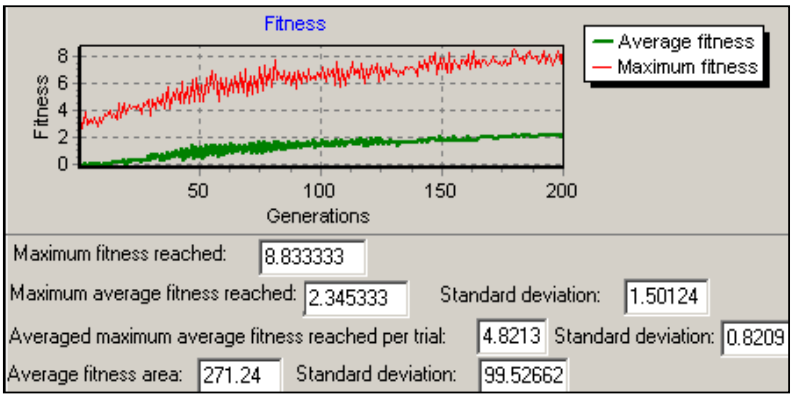


Figure 7-8 The Darwinian evolution and reinforcement learning model's performance in an environment with fitness consequence alternations after every generation.

Figure 7-8 shows that our agents now have the ability to effectively adapt to the most unstable conditions. Although fitness consequence reversals are occurring after every generation, with food becoming poison and vice-versa, the above performance demonstrates a consistent, stable growth in fitness levels, not suffering from the wild oscillations of the evolution model or the non-existent growth of the reinforcement learning model. As a result of combining the two processes, an adaptive ability is produced that either alone does not posses.

An evaluation of the agent's innate, genotype knowledge reveals further insights into the model's adaptive nature.

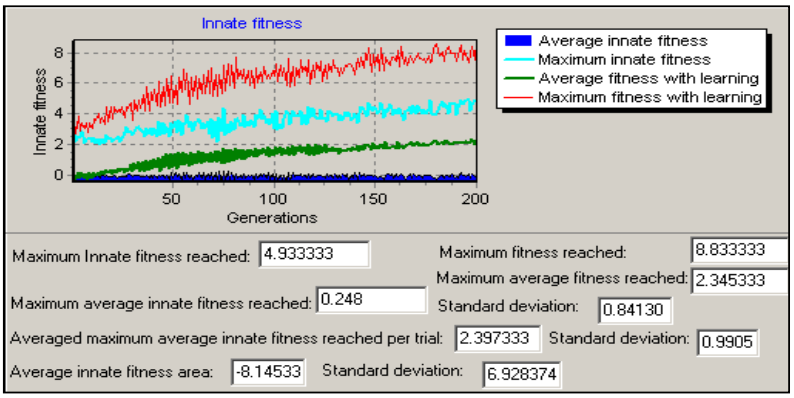


Figure 7-9 The Darwinian evolution and reinforcement learning model's innate and learning performance in an environment with fitness consequence alternations after every generation.

As Figure 7-9 indicates, the agents' innate knowledge, when evaluated in isolation from learning, does not demonstrate any growth in effectiveness. In fact, it maintains

a consistent evaluation of around 0. However, it is this knowledge, evolving in the agents' genotypes that allows them to produce an improving performance, since all learnt knowledge is lost at the end of every generation. It appears then, that the knowledge evolved in the agent's genotype is useful in terms of its ability to complement learning rather than follow it and, as a result, produce effective solutions to a difficult situation.

In addition, the evolution process was found to be effective in its ability to optimize reinforcement feedback values, thereby further complementing the ability of learning by adjusting reinforcement levels according to the environment's degree of instability.

In stable environments the evolution process produces innate agent knowledge that needs only small reinforcements to result in improvements, since the evolution process is in itself capable of adapting to such environments with learning playing a smaller role. However, in unstable environments the evolution model produces knowledge of a general nature that requires large reinforcements to transform that knowledge into the appropriate solution

The following demonstrates the model's ability to adjust the initial feedback (0.4, -0.4 and –0.02 for locating food, poison and per action, respectively) to an appropriate level required for an effective adaptation to the most unstable conditions.

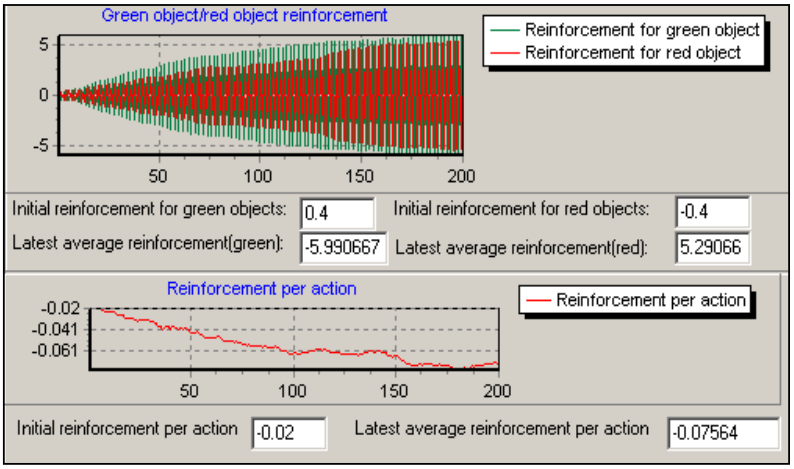


Figure 7-10 Optimizing reinforcement feedback values using our Darwinian evolution and reinforcement learning model in the most unstable environment.

Note that the reinforcement feedback values utilized for locating green and red objects (Figure 7-10) are reversed after every generation, consistent with the fitness consequence alternations. This occurs as the reinforcement values, like the fitness consequences, are associated with the objects according to their food or poison nature. Since the green and red objects' food or poison nature is reversed after every generation, so are their associated reinforcements.

As the reinforcement values associated with each object are contained in each agent's description, they are open to optimization. The above graph demonstrates that as a

result of the optimization process, the reinforcement values associated with locating food and poison are consistently adjusted towards larger values over the course of generations. It also appears that a greater degree of negative reinforcement per action is required. These adjustments, then, are an evolved response to the degree of instability being experienced by the agents and clearly indicate that a greater degree of learning is required in response to increasing instability. Recall our comments regarding the ineffectiveness of the model utilizing small, fixed reinforcement values. The degree of reinforcement utilized in the stable environment may not be appropriate with regard to increasing instability. Optimization, then, demonstrates that it can provide an effective mechanism of adjusting such parameters according to the requirements of the environment.

Thus, our evolution process can complement reinforcement learning by providing both innate, general knowledge regarding the population's experience of the environment and appropriately adjust the degree of reinforcement learning required in order to modify the agents' evolved knowledge according to the particular requirements of their environment.
The following section provides a more detailed investigation regarding the nature of the evolved knowledge and its interaction with learning.

7.3.2 Adaptive Behaviour and Neural Network Knowledge

In the stable environment behaviour and the neural network knowledge that produces it, is similar to that of the evolution model. However, should an agent eat poison then such behaviour can be immediately negatively reinforced, reducing the probability of it occurring throughout the remainder of the agent's lifetime. This can explain the lower evaluations of innate knowledge that were recorded. That is, innate knowledge is evolved taking it for granted that behaviour such as eating poison will be negatively reinforced. When evaluated without learning, such knowledge is not appropriately modified and, thus, it receives a lower evaluation than fixed, innate knowledge evolved without the presence of learning. This emphasizes our previous statement regarding the evolution of innate knowledge relying on learning to take place and, thereby, potentially hindering the process.

However, learning has clearly been shown to be useful under conditions of instability. Thus, this section is devoted to extracting the nature of our Darwinian evolution and reinforcement learning model's adaptive ability with regard to the agents' neural network knowledge, both evolved and learnt, responsible for their effective behaviour in conditions of instability. The following figures represent an agent's innate and learnt neural network information and its influence on behaviour in the 200th generation of a simulation utilizing the most unstable environment, with fitness consequence reversals occurring after every generation.

We begin by extracting an agent's innate knowledge regarding green objects.

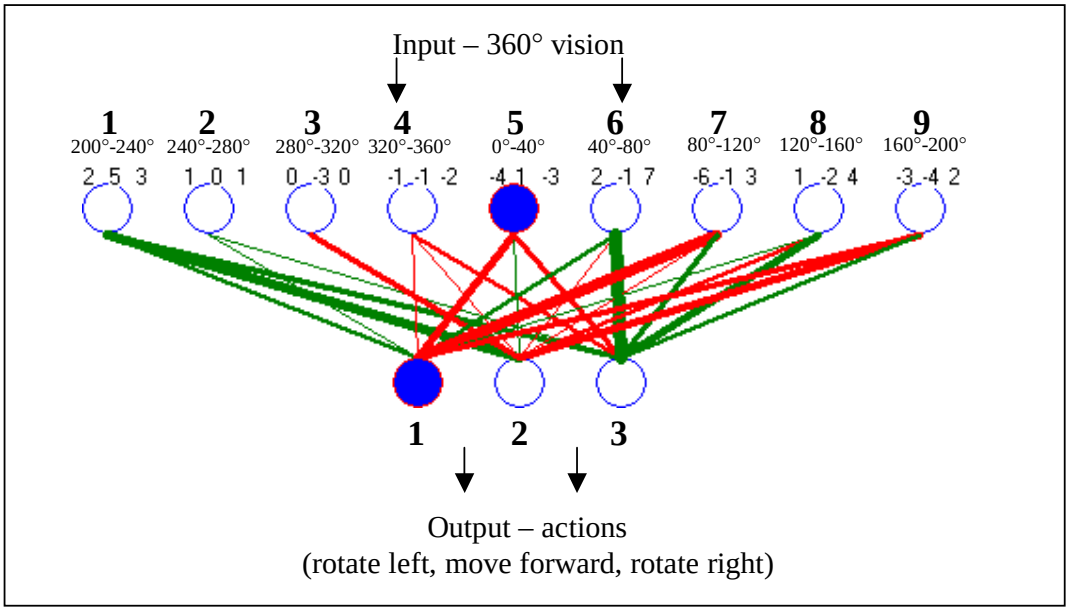


Figure 7-11 An agent's neural network containing innate information responsible for mapping visual input regarding green objects to an output action.

Figure 7-11 represents an agent's innate neural network knowledge used to map visual input regarding green objects to an output action. It translates into the following figure indicating the effect that the position of green objects has on the agents' actions (Figure 7-12). Remember, the agent's innate knowledge cannot be evolved to determine whether green or red objects are food or poison, since both objects' fitness consequences are reversed after every generation. An evolution process, learning at a generation timescale, is limited in its ability to effectively extract such a fast changing regularity. However, it can still produce useful knowledge with regard to such changes.

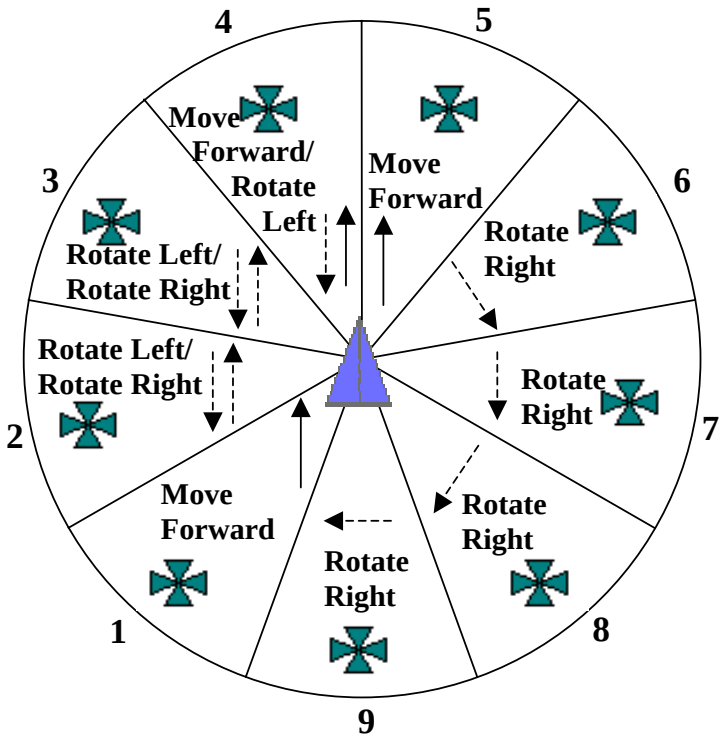


Figure 7-12 The influence of green objects on an agent's initial behaviour.

The above figure indicates that our agent contains innate behaviour to locate green objects, regardless of their fitness consequences. All mappings essentially comply with this strategy. Should the object occur in regions 6,7, 8 or 9 then right (clockwise) rotations will be chosen, orientating the agent towards the object (region 5) from where it can move forward and locate it. Note that the forward movements of region 1 will place the object in region 9, thereby, also resulting in the above behaviour. Should the object occur in region 4 then the forward movements and leftward rotations will also effectively locate it. Regions 2 and 3 are the only regions that appear to not be directly involved with this strategy. However, the left and right actions associated with these regions will have the ultimate result of placing the object in either region 4 or 1 from where it can be easily located. In addition, the influence of the red object's position will also likely resolve any stagnation in the agent's behaviour.

With regard to red objects the agent demonstrates similar behaviour.

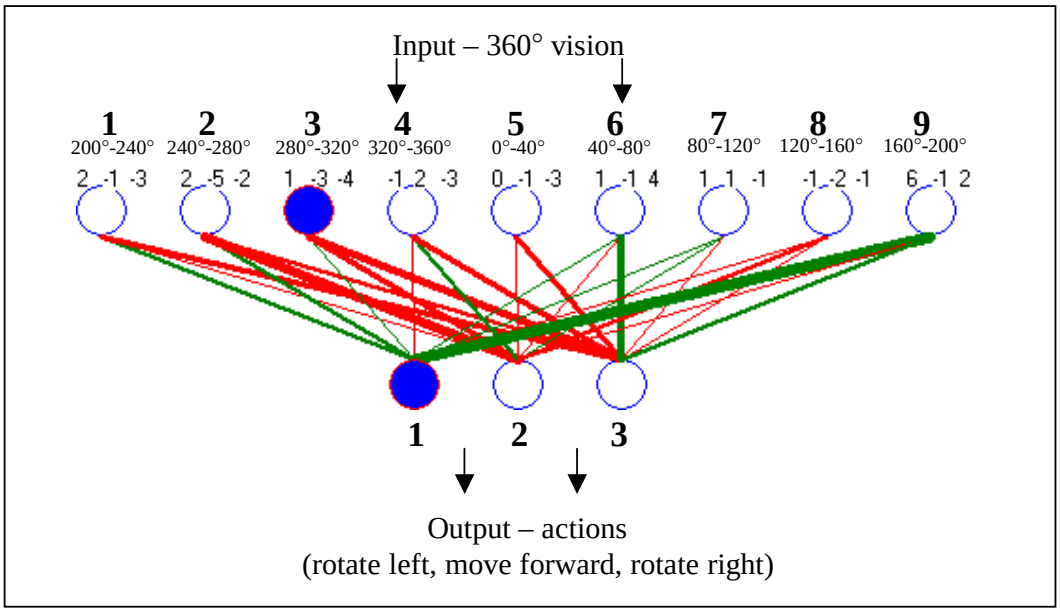


Figure 7-13 An agent's neural network containing innate information responsible for mapping visual input regarding red objects to an output action.

Figure 7-13 represents an agent's innate neural network knowledge used to map visual input regarding red objects to an output action. It translates into the following figure (Figure 7-14) indicating the effect that the position of red objects has on an agent's initial behaviour.

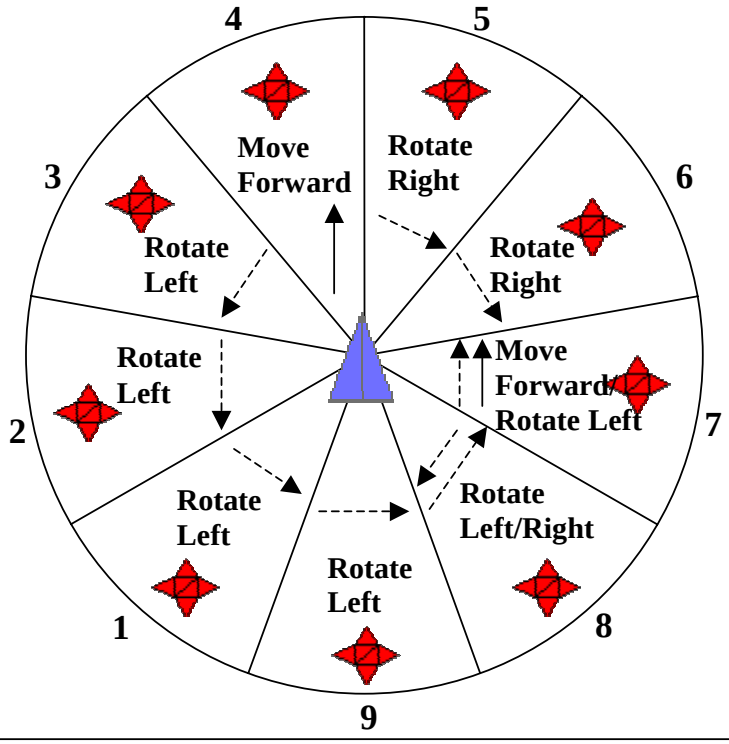


Figure 7-14 The influence of red objects on the agent's initial behaviour.

The left (anti-clockwise) rotations of regions 3, 2, 1 and 9 will orientate the agent towards the red object (region 4) from where it can move forward and locate it, with the right (clockwise) rotations of region 5 and 6 having the same effect. The agent's response to the red object occurring in region 7 will be to either move forward or rotate left, both actions bringing the object to region 8. This is the only indecisive region with the agent equally likely to rotate left or right. However, it is unlikely that the red object will remain in this region for any length of time, since the influence of the green object's position is likely to resolve such situations.

It appears, then, that knowledge has been evolved to complement the ability of our agents to discriminate between green and red objects according to their associated reinforcements. Instead of avoiding one object and locating the other, the agent's genotype knowledge was altered to produce behaviour that demonstrates a tendency to locate both objects. Individual reinforcements can then be used to appropriately modify this behaviour according to the nature of the objects, food or poison.

The following figures demonstrate how the above agent's knowledge is altered in response to learning.

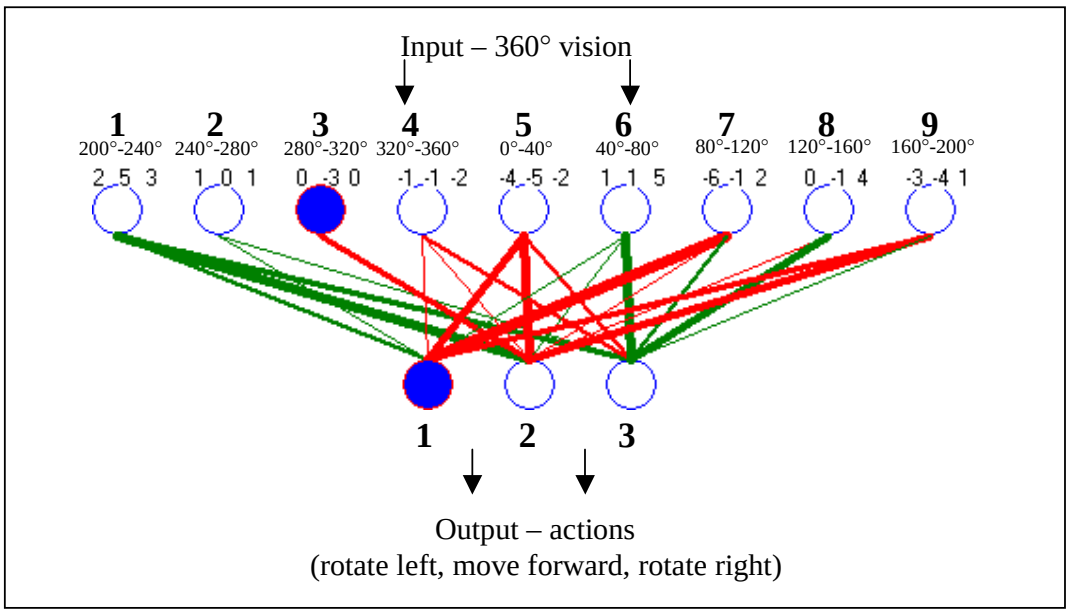


Figure 7-15 An agent's neural network responsible for mapping visual input regarding green objects to an output action – after learning.

Figure 7-15 represents an agent's neural network knowledge, after learning has occurred, used to map visual input regarding green objects to an output action. It translates into the following figure indicating the effect that the position of green objects now has on the agent's actions (Figure 7-16).

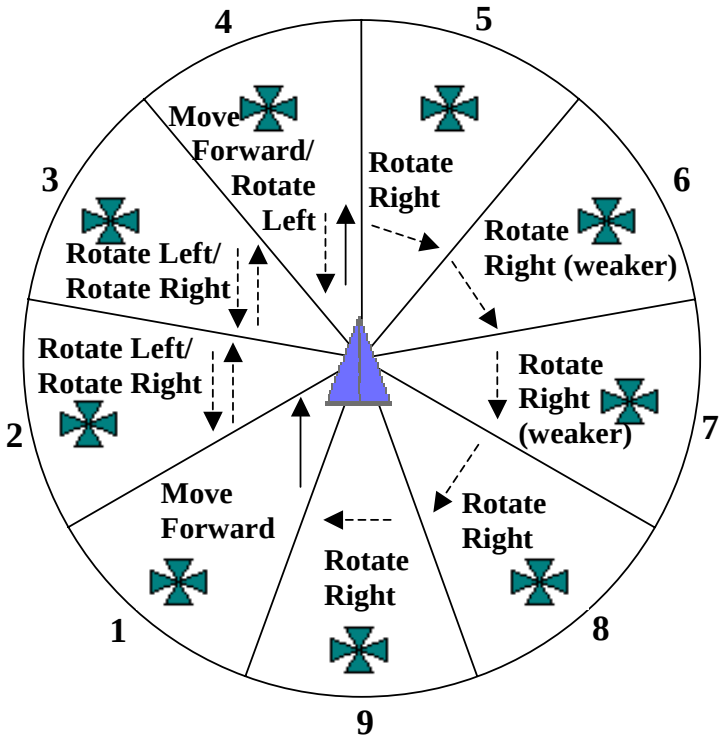


Figure 7-16 The influence of green objects on the agent's behaviour after learning.

As a result of learning, negative reinforcement due to the agent locating a green object, that is poison in this particular generation (200th), the agent is no longer inclined to move forward should the object occur in region 5. Note the strong inhibitive connection of –5 from input neuron 5 to the move forward neuron (neuron 2) shown in Figure 7-15. The tendency to rotate right, associated with regions 6 and 7, has also been weakened, since this is also behaviour that was involved in the green object being located. Thus, the agent is less likely to demonstrate behaviour resulting in green objects (poison) being located.

Figure 7-17 represents the agent's neural network knowledge, after learning has occurred, used to map visual input regarding red objects to an output action.

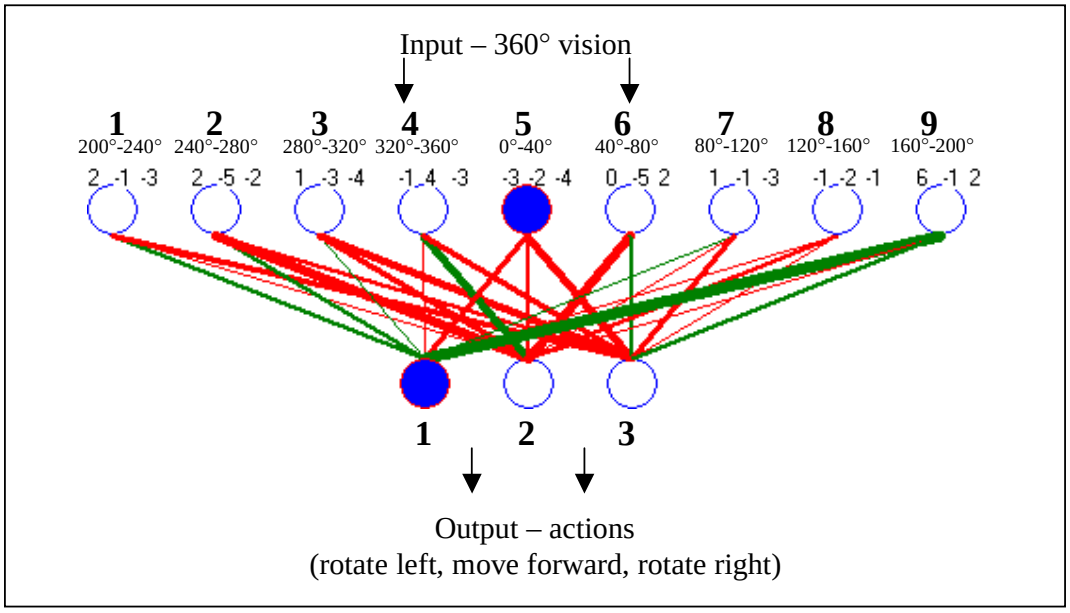


Figure 7-17 The agent's neural network responsible for mapping visual input regarding red objects to an output action – after learning.

It translates into the following figure (Figure 7-18) indicating the effect that the position of red objects now has on the agent's behaviour.

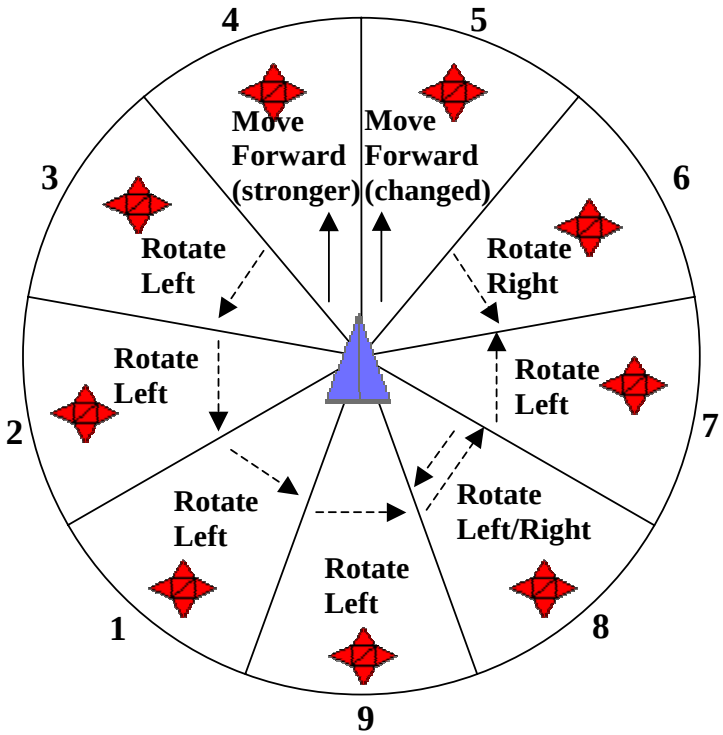


Figure 7-18 The influence of red objects on the agent's behaviour after learning.

As is shown by Figure 7-18 the agent has become more inclined, due to positive reinforcement, to move forward should the red (food in the 200th generation) object occur in the two regions associated with its frontal vision.

Thus, the agent's initial, exploratory behaviour was successfully modified, via positive and negative reinforcements, to allow it to effectively discriminate between food and poison objects throughout the remainder of its lifetime.

Our evolution process, operating on the genotype to produce useful exploratory knowledge, and our reinforcement learning process, utilizing and refining the knowledge, provides an effective complementary relationship between evolution and learning in order to solve a difficult problem that requires the alternating application of two different solutions throughout the simulation's duration. Since our Darwinian evolution and reinforcement learning model utilizes learning at two different levels, our agents can assimilate knowledge regarding both required solutions, via evolution, that can then be activated, via lifetime learning, according to the requirements of their environment.

Due to the fact that learnt knowledge is not inherited by offspring, providing a separation between genotype and phenotype knowledge, our agents can effectively discriminate between food and poison objects within their individual lifetimes in an environment demonstrating fitness consequence reversals after every generation.

Thus, our evolution processes ability to utilize generations of agents' experience of the environment can complement learning by producing general, genotype knowledge regarding the changes in the environment that can then be further refined according to the needs of an agent in its particular generation. As a result, the population's general experience of the environment, over generations, and the individual's ability to learn, within its lifetime, are combined to produce an adaptive quality that either process alone does not posses.

7.4 Conclusion

Our hybrid Darwinian evolution and reinforcement learning model combines the two previous models to produce a mechanism capable of learning at both a population and an individual level. Thus, learning occurs at a genotype level (via evolution) and phenotype level (via reinforcement learning). This allows regularities at two different timescales to be extracted, the genotype encoding information influenced by the experience of the population over many generations and the phenotype encoding information influenced by the experience of an individual during its lifetime. As a result knowledge relating to the general characteristics of the environment over many generations can be maintained in the population's genotypes with phenotype (reinforcement) learning being utilized to adapt a particular agent to the particular characteristics of its environment.

It is shown that the Darwinian evolution and reinforcement learning model makes use of the separation of learning at a genotype and phenotype level to produce agents that can effectively deal with the most unstable environments. An analysis of the genotype reveals it to contain general information that guides the agent's learning ability. It is

the innate information contained in the agent's genotype and produced by an evolution process that allows the agents to effectively utilize their individual reinforcement learning ability to discriminate between food and poison objects.

The nature of this knowledge is to produce agents that initially attempt to locate both objects, after which they are appropriately reinforced, thereby effectively learning the relationship between an object and its fitness consequences. This relationship is then "forgotten" in the following generation, with offspring agents returning to their innate knowledge. The advantage being that knowledge, now possibly incorrect, from a previous environment does not need to be unlearnt. Rather, innate knowledge that has been formed from the population's experience of many environments is utilized to provide the new agents with a general knowledge of their environment that can be further refined through reinforcement learning.

The disadvantage of this model is that it is shown to be less effective in the stable environments than the reinforcement learning model and only equivalent to the evolution model if fixed reinforcements are used, while being a bit worse if the reinforcements are open to optimization. Attempting to optimize reinforcement values is found to be necessary for an effective performance if the environment decays from stable to unstable, since different degrees of learning are required in environments of a different stability. This is found to be problematic in a stable environment as it extends the solution space to a degree that is difficult for the search process to effectively handle. The solution space terrain becomes more difficult to navigate, hampering the search process's ability to find solutions closer to the optimum.

However, with a view towards adapting in an environment of changing stability it is shown that optimizing the feedback values is useful and necessary, as the degree of reinforcement required varies according to the stability of the environment. In unstable environments the evolution model produces knowledge of a general nature that requires large reinforcements to transform that knowledge into the appropriate solution. In stable environments the evolution process produces innate agent knowledge that needs only small reinforcements to result in improvements, since the evolution process is in itself capable of adapting to such environments with learning playing a smaller role.

Allowing the system to optimize reinforcements provides it with the ability to adapt the degree of individual learning required according to the demands of the environment. As a result, our evolution process can complement reinforcement learning by providing both innate, general knowledge regarding the population's experience of the environment and appropriately adjust the degree of reinforcement learning required, thereby modifying the agents' evolved knowledge according to the particular requirements of their environment.

Thus, optimizing reinforcement values in conjunction with a separation of learning at a genotype and phenotype level provides a model that can effectively produce solutions across the range of environments from stable to unstable.

Chapter 8: Design and Evaluation of a Lamarckian Evolution and Reinforcement Learning Model

8.1 Introduction

Under a Lamarckian framework an individual's learnt knowledge is inherited by its offspring. In nature Lamarckian evolution would be very difficult to achieve, learnt knowledge would need to be genetically encoded. However, for our purposes Lamarckian evolution is a viable problem solving mechanism.

Recall that when we investigated reinforcement learning our agents did not die or reproduce, but simply lived and learnt until the end of the simulation. Learnt knowledge was not lost, but only altered by further learning. The difference with Lamarckian evolution is that it has the ability to reproduce (amplify) fitter solutions, produced by learning, and remove weaker solutions. Thus, learnt knowledge is selected and reproduced, in contrast to our previous evolution models' reproduction of genotype knowledge produced as a result of mutations.

Learning, then, can be considered another genetic operator, as it has the ability to directly modify knowledge that will be inherited by an agent's offspring. As a result there is no longer a separation of learning at a genotype and phenotype level. Learnt knowledge is maintained and strengthened via the processes of selection and reproduction. Thus, the individual's experience of the environment and the knowledge it gains has a far stronger influence on the population over the generations.

The result is that useful knowledge that is consistent with an individual's experience of the environment and the population's experience over generations can be far more effectively discovered. However, such combinations can also compromise the global and local search characteristics of evolution and learning. That is, an over emphasis on local search (the individuals' learning ability) can be detrimental to global search characteristics (the population's learning ability as a result of evolution), leading to a premature convergence of the population towards learnt solutions. In addition, should the experience of the individual differ markedly in nature from that of the population's previous experience over many generations then the ability to maintain useful, general knowledge regarding the environment may be compromised.

Thus, it is our objective to examine how effective such characteristics of a Lamarckian evolution and reinforcement learning model are in adapting to an environment of increasing instability.

The following presents our:

- Design of a Lamarckian evolution and reinforcement learning model with a view towards maximizing its ability to produce adaptive agents in an initially stable environment that becomes increasingly unstable.
- Evaluate the effectiveness and adaptive nature of the model with a view towards extracting the advantages and limitations of such models in dealing with stable environments that become increasingly unstable.

8.2 Design

Our agents already have all the necessary attributes to achieve Lamarckian evolution. All that is required is that our offspring agents' learnable weight values be set equal to their parents' learnt weight values. Reinforcement learning is then applied to these weight values and the process repeated. Thus, offspring start off life from where their parents ended it, except they may also be further mutated.

With regard to optimising reinforcement feedback parameters, this is done as before. The parameters are included in the agent's description and altered via mutations. Those agents that contain the appropriate values, to the benefit of learning and, thus, the agent's fitness, accrue greater probabilities of selection and reproduction. As a result of such selective pressure the reinforcement feedback parameters can be moved towards more effective values (refer to appendix L for data concerning the appropriate degree of change – ±0.7 for food/poison feedback and ±0.005 for feedback per action was utilized for this model). Thus, our Lamarckian evolution and reinforcement learning model can also adjust the degree of learning required according to the nature of the simulation.

This model, then, is essentially equivalent to the previous model, except that the offspring agents inherit their parents' learnt knowledge. However, this learning is of a different nature and, therefore, requires different feedback parameters in order to maximize the model's effectiveness.

8.2.1 Simulation Framework and Parameters

Our Lamarckian evolution and reinforcement learning model is applied utilizing the same simulation framework as our previous models.

The important constants are:

- A generation lifetime of 500
- A simulation length of 200 generations
- A population of 25
- 1 food object in the world at any time
- 1 poison object in the world at any time
- 30 trials per sample

As was the case with the Darwinian evolution and reinforcement learning model, several options need to be explored to effectively evaluate our Lamarckian model. We can use fixed reinforcement values in conjunction with a fixed mutation rate, an individual mutation rate that is allowed to self-optimize or no individual neural network mutations at all, since reinforcement learning can now be considered a genetic operator. We can also allow the reinforcement values to be adjusted in conjunction with the previous options.

However, in order to choose effective fixed reinforcement feedback values we utilized no neural network mutations at all, since it was considered likely that the undirected neural network mutations would not be of any benefit to the directed process of reinforcement learning. In fact, our experiments evaluating the model using no neural

network mutations, using a fixed degree of mutations and allowing this degree to be optimized do indicate that mutations are no longer of any use in producing effective neural network knowledge. Having being overridden by the reinforcement learning process.

Based on the data of appendix K, the fixed reinforcement feedback values used are: 3, -3 and –0.019 for locating food, poison and per action, respectively. These are used in conjunction with a fixed mutation rate, an individual mutation rate that is allowed to self-optimize and no individual neural network mutations at all, in order to evaluate our Lamarckian evolution and reinforcement learning model's response to an initially stable environment that becomes increasingly unstable. Experiments that adjust the reinforcements in conjunction with fixed mutation rates, an individual mutation rate that is allowed to self-optimize and no individual neural network mutations at all were also performed. Refer to appendix M for complete data regarding these experiments.

The most successful of these options – using no mutations and fixed reinforcements of 3, -3 and –0.019 for locating food, poison and per action, respectively, were utilized to evaluate our Lamarckian evolution and reinforcement learning model.

8.3 Evaluation of Our Lamarckian Evolution and Reinforcement Learning Model

8.3.1 Performance in Stable to Unstable Environments

8.3.1.1 Overview

The following presents our Lamarckian evolution and reinforcement learning model's performance in response to differing levels of environmental stability, ranging from stable to highly unstable.

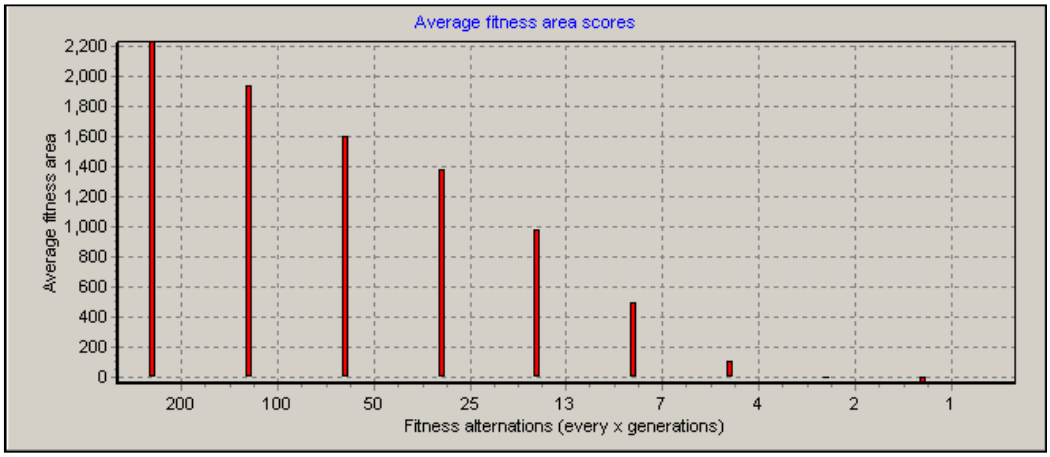


Figure 8-1 Average fitness area scores produced by our Lamarckian evolution and reinforcement learning model in stable to highly unstable environments.

As before, average fitness area scores are presented, since they are considered to be the most informative in terms of consistency of performance. Refer to appendix M for complete data regarding the other performance measures.

As shown by Figure 8-1, our Lamarckian evolution and reinforcement learning model's performance does decay into ineffectiveness in response to the most unstable environments, beyond fitness consequence reversals every 7 generations. However, note the high average fitness area scores produced in response to the more stable conditions. They are all greater than that of the Darwinian evolution and reinforcement learning model, with the average fitness area levels produced in the most stable environment being a full 1000 points greater. For a more complete comparison refer to chapter 10.

It appears, then, that the model can effectively deal with the increasing instability to a large degree, but ultimately its performance rapidly declines into ineffectiveness.

The following provides details regarding the model's performance in response to several different degrees of environmental stability.

8.3.1.2 Performance Details

8.3.1.2.1 Stable Environment

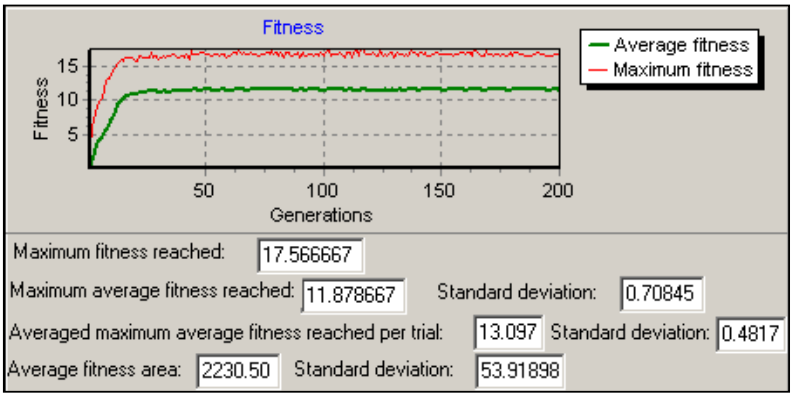


Figure 8-2 The Lamarckian evolution and reinforcement learning model's performance in the stable environment.

It is immediately apparent that the learning speed of this model is highly effective in the stable environment (Figure 8-2). Within a few generations maximum fitness levels are being produced that are greater than those of the evolution and Darwinian evolution and reinforcement learning model over the entire course of their simulation. The reinforcement learning model comes close to these levels, but only after about 75 generations have elapsed. Clearly, selection and reproduction of learnt knowledge, produced as a result of reinforcements, can dramatically improve adaptation time in response to the stable environment.

8.3.1.2.2 Fitness Consequence Alternations Every 25 Generations

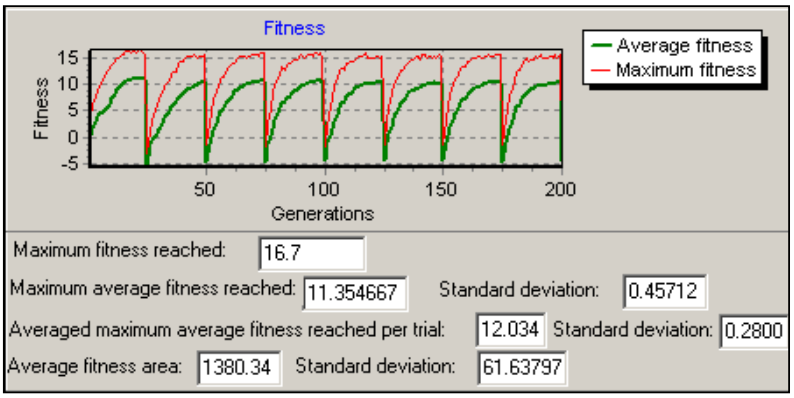


Figure 8-3 The Lamarckian evolution and reinforcement learning model's performance in an environment with fitness consequence alternations every 25 generations.

Increasing the level of instability in the environment, by utilizing fitness consequence reversals every 25 generations (Figure 8-3), does not appear to hinder the model's ability to produce effective fitness levels. Note that, even though offspring agents inherit incorrect solutions, these are quickly unlearnt and the correct solution relearnt within a few generations to produce peak fitness levels almost equivalent to those achieved in the stable environment. Although the changes produce a large drop in fitness levels, this is quickly remedied by the models rapid unlearning/relearning time.

8.3.1.2.3 Fitness Consequence Alternations Every 7 Generations

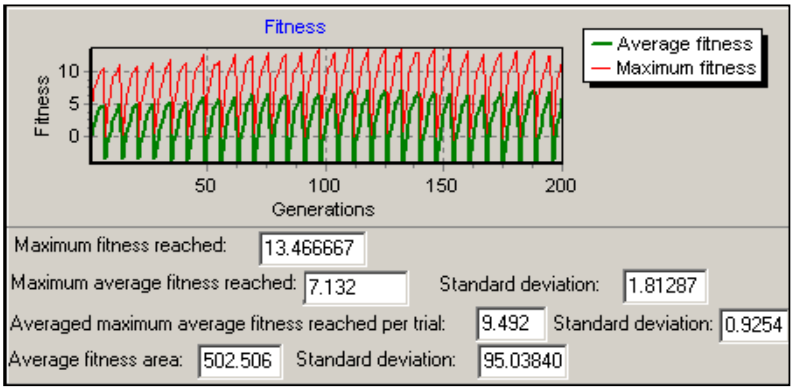


Figure 8-4 The Lamarckian evolution and reinforcement learning model's performance in an environment with fitness consequence alternations every 7 generations.

At a level of instability utilizing fitness consequence reversals every 7 generations (Figure 8-4), the model still maintains its ability to produce high peak fitness levels greater than those of the other models in the equivalent environment. However, the performance of the model is now of a highly oscillating nature. Although still producing a high average fitness area score, indicating the degree to which food is being consumed as compared to poison over the full course of the simulation, it does not demonstrate the stability of the Darwinian evolution and reinforcement learning model. The Lamarckian evolution model is limited by short periods of ineffectiveness after every fitness consequence reversal, since it cannot effectively assimilate knowledge regarding such changes and thereby limit their impact. Instead, offspring inherit incorrect solutions resulting in negative fitness scores being produced. In addition, the slightly improving peak average fitness levels of Figure 8-4 occur at the cost of producing larger negative average fitness scores. Thus, rather than adapting to increasing instability, it is the model's effective unlearning/relearning ability that produces its high fitness measures. However, since the model cannot assimilate the environment's changes, it produces a highly oscillating performance unlike the previous model's ability to produce consistently increasing positive fitness scores in response to the same conditions. These limitations of the model's adaptive nature become more apparent in response to increasing levels of instability.

8.3.1.2.4 Fitness Consequence Alternations After Every Generation

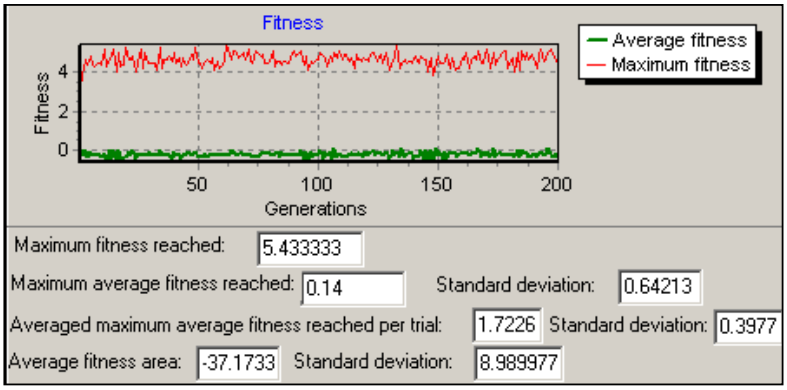


Figure 8-5 The Lamarckian evolution and reinforcement learning model's performance in an environment with fitness consequence alternations after every generation.

In response to the most unstable conditions the model's performance is completely ineffective, producing the worst average fitness area score (-37.1733) of all the models (Figure 8-5). At this level of instability the fitness consequence reversals are occurring at a frequency that is beyond the capabilities of the model's unlearning/relearning effectiveness. Thus, the inheritance of learnt knowledge, overemphasising the experience of particular agents in a particular generation, has ultimately become disruptive to the their adaptive ability. Although highly effective under stable conditions and moderate levels of instability, the Lamarckian model's effectiveness is limited in response to the most unstable conditions.

In contrast, the Darwinian model maintains both the population's ability to extract knowledge regarding the environment, over many generations, and the agents' individual ability to learn, within their lifetime, to produce a learning process that can effectively adapt to the most unstable conditions. Recall the constant, stable growth demonstrated by that model in response to this level of instability.

Thus, under a Lamarckian framework the population's general experience of the environment can be compromised by that of the individual's. In stable environments, where the individual's experience is consistent with that of the population's, this adaptive nature of Lamarckian evolution can prove beneficial. However, in response to increasing levels of change it can ultimately become a disadvantage.

8.3.2 Adaptive Behaviour and Neural Network Knowledge

Our Lamarckian agents' behaviour and neural network knowledge is very much like that of the reinforcement learning model, since it makes use of the same fundamental learning operator. However, due to the processes of selection and reproduction, correctly reinforced agents are now produced at a much faster rate, i.e., within 15 generations as compared to about 75 generations using the reinforcement learning model. As a result, the Lamarckian model can cope more effectively with increasing levels of instability. Ultimately, though, an equivalently ineffective performance is produced in the most unstable environment.

8.4 Conclusion

Our Lamarckian evolution and reinforcement learning model is shown to demonstrate adaptive characteristics that are highly effective in response to the stable environments. Selection and reproduction combined with reinforcement learning creates a model that has the ability to utilize useful knowledge produced by reinforcements, as opposed to random mutations, to accelerate the search process. Since mutations are undirected, a superior performance in stable environments can be produced using reinforcement learning as a more informative genetic operator. This also results in the Lamarckian evolution and reinforcement learning model demonstrating an exceptional ability to unlearn incorrect knowledge and relearn correct solutions in response to changes in an environment's characteristics. However, unlike the Darwinian model, it is incapable of assimilating knowledge regarding such changes in order to improve its effectiveness over the course of generations. As a result, its high levels of effectiveness ultimately and rapidly decay in the environments utilizing the highest levels of instability.

As is the case with the evolution model and the reinforcement learning model, our Lamarckian evolution and reinforcement learning model is fundamentally flawed in dealing with highly unstable environments. Solutions are produced that need to be unlearnt as soon as the environment changes. Therefore, our Lamarckian evolution and reinforcement learning model is unable to adapt to the most unstable environments tested while being highly effective in dealing with the more stable environments.

Since our previous conclusions indicate that Darwinian and Lamarckian evolution models have adaptive characteristics that are ideally suited towards particular situations, work was performed to investigate the potential of combining both forms of evolution. The following presents our design of hybrid Darwinian/Lamarckian models that combine the advantages of both Lamarckian and Darwinian evolution with the objective of producing a model demonstrating greater adaptability to both stable environments and in response to increasing levels of instability.

Chapter 9: Design and Evaluation of Hybrid Darwinian/Lamarckian Evolution and Reinforcement Learning Models

9.1 Introduction

Hybrid Darwinian/Lamarckian evolution model can be designed to incorporate degrees of inheritance or non-inheritance of learnt knowledge in the population's agents. Related work, previously described in section 3.8, has been performed to investigate the usefulness of such agents and populations that are partially Lamarckian and Darwinian. Work by Sasaki and Tokoro [Sasaki and Tokoro, 1999] utilized agents incorporating an heredity rate that determines the degree to which they inherit the learnt knowledge of their parents. It was found that fully Lamarckian agents were the most successful in adapting to stable environments while fully Darwinian agents were the most successful under dynamic conditions. Other work by Houck, Joines, Kay and Wilson [Houck, Joines, Kay and Wilson, 1997] utilized a heterogeneous population of both Lamarckian and Darwinian individuals to provide a certain degree of inheritance and non-inheritance in the population. For example, 50% of the population could be Lamarckian while the other 50% is Darwinian. Various rates of Lamarckism and Darwinism were investigated, ranging from completely Darwinian populations to completely Lamarckian populations, in response to a number of test problems. This investigation found that a strategy utilizing a 20% to 40% partially Lamarckian population produced the most successful performances across the entire set of test problems, thus indicating the potential benefits of such hybrid strategies.

Our approach investigates the use of four different strategies combining Lamarckism and Darwinism, emphasizing the ability to optimize the degree of inheritance occurring in the population according to the requirements of the environment. Since the effectiveness of differing degrees of inheritance depends on the nature of the environment, it is advocated that such methods can improve adaptability in response to environments of changing stability.

The following presents our:

- Design of several hybrid Darwinian/Lamarckian evolution and reinforcement learning models with a view towards maximizing their ability to produce adaptive agents in an initially stable environment that becomes increasingly unstable.
- Evaluation of the effectiveness and adaptive nature of the models with a view towards extracting the advantages and limitations of such models in dealing with stable environments that become increasingly unstable.

9.2 Heterogeneous Population Model

Our first model utilizes a heterogeneous population of both Lamarckian and Darwinian agents in conjunction with the ability to mutate the agents' Darwinian or Lamarckian nature. That is, Darwinian agents can be mutated to become Lamarckian agents and vice-versa. As a result, the degree of inheritance or non-inheritance can be optimized according to the characteristics of the environment.

9.2.1 Design

The requirements of our model are:

- A heterogeneous population of agents.
- The ability to optimize the Lamarckian and Darwinian nature of the population according to the characteristics of the environment.

9.2.1.1 Heterogeneous Population of Agents

Our hybrid model utilizes a population consisting of both Darwinian and Lamarckian agents. To achieve this each agent is assigned an extra attribute (gene) defining the nature of its inheritance. This attribute is utilized to initialize the population to be either 100% Darwinian, 100% Lamarckian or a mixture of both by randomly initializing each agent as either Darwinian or Lamarckian.

At the end of each generation our model examines the attribute to determine whether a selected agent is Lamarckian and should produce offspring that inherits their learnt knowledge or whether it is Darwinian and should produce offspring that inherits the knowledge contained in their genotypes. Thus, as a result of selection, Darwinian or Lamarckian agents can be reproduced in response to the adaptive requirements of the environment. This is in contrast to the previously mentioned work that maintains a fixed level of Darwinian and Lamarckian evolution in the population. Our approach provides the ability to optimize the degree of either type of evolution according to the success of the agents in response to the environment's conditions.

9.2.1.2 Optimization

Optimization is achieved by allowing the agents' Darwinian/Lamarckian genes to be mutated. Should an agent be chosen for mutation, with a probability determined by the population mutation rate, then the agent is designated a 1/population size probability of being changed to its converse evolutionary type. Since the utilized population mutation rate is quite high (0.8), this is sufficient to maintain at least one Darwinian or Lamarckian agent in the population at any time. This ensures that diversity can be maintained and, thus, either form of evolution is always available for selection should circumstances require change.

9.2.1.3 Simulation Framework and Parameters

Our hybrid Darwinian/Lamarckian heterogeneous population model is applied utilizing the same simulation framework as our previous models.

The important constants are:

- A generation lifetime of 500
- A simulation length of 200 generations
- A population of 25
- 1 food object in the world at any time
- 1 poison object in the world at any time
- 30 trials per sample

The following evaluation utilizes agents that are randomly initialized as either Darwinian or Lamarckian, thereby providing a population that initially consists of roughly 50% of either type. Thus, the population is not biased towards either a Lamarckian or Darwinian nature, allowing it to be quickly optimized towards the required degree of inheritance appropriate to the conditions of the environment – from stable to unstable.

Reinforcement feedback values of 3, -3 and –0.019 are utilized for locating food, poison and per action, respectively. These are equivalent to those of the Lamarckian model, found to be highly successful in the initially stable environment. Experiments were conducted to evaluate the benefits of allowing these values to be adjusted, should the more unstable environments require different feedback values. However, optimization was found to be unnecessary, since these fixed values are highly effective in the stable environment and in response to increasing levels of instability, complementing both the Lamarckian agents ability to adapt to the sable environment and the Darwinian agents ability to adapt to the unstable environments.

Optimizing the individual's mutation rate was also found to be unnecessary, slightly superior results being produced using a population mutation rate of 0.8 and a fixed individual mutation rate of 0.1. Refer to appendix N for complete data regarding these experiments.

The following, then, presents an evaluation of our hybrid Darwinian/Lamarckian evolution and reinforcement learning model utilizing fixed reinforcement values of 3, -3 and –0.019 for locating food, poison and per action, respectively, with a population mutation rate of 0.8 and a fixed individual mutation rate of 0.1. The population's agents are randomly initialized as either Darwinian or Lamarckian, providing an initial heterogeneous population that can then be adjusted according to the requirements of the environment.

9.2.2 Evaluation of Our Heterogeneous Population Model

9.2.2.1 Performance in Stable to Unstable Environments

This model proves to be highly useful in adapting to environments that undergo changes in stability.

9.2.2.1.1 Overview

The following presents our hybrid Darwinian/Lamarckian heterogeneous population model's performance in response to differing levels of environmental stability, ranging from stable to highly unstable. Refer to appendix N for complete data regarding this evaluation.

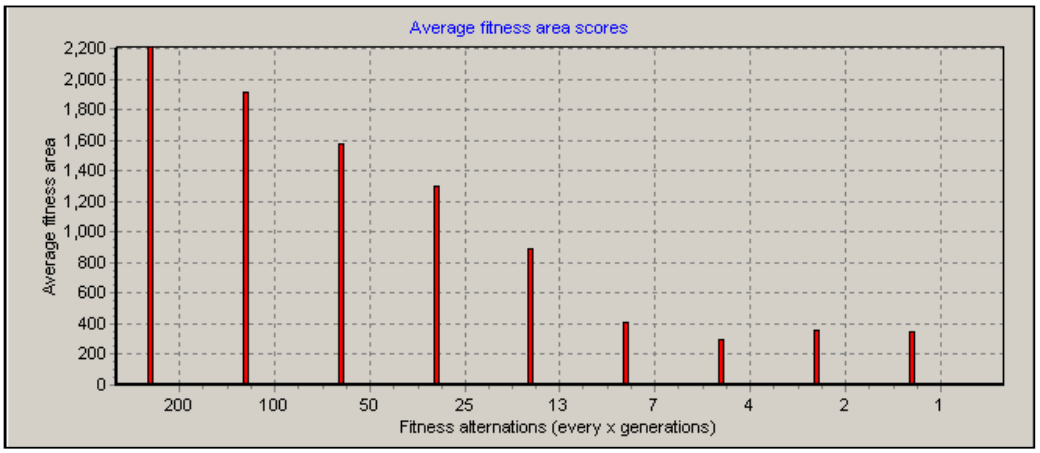


Figure 9-1 Average fitness area scores produced by our hybrid Darwinian/Lamarckian heterogeneous population model in stable to highly unstable environments.

As shown by Figure 9-1, this model is highly effective in response to all levels of stability. Its performance across the range of environments is equivalent to that of the best performances produced by the Darwinian and Lamarckian models. This is possible as the model combines the advantages of both Lamarckian and Darwinian evolution by utilizing a heterogeneous population that can be optimized according to the requirements of the environment. Thus, it can appropriately adapt to both stable and unstable conditions.

Note that the above performance levels are typically Lamarckian in nature until a level of instability utilizing fitness consequence alternations every 7 generations has been reached. Beyond this level of instability the performance levels become Darwinian in nature – recall the Lamarckian models inability to adapt to the most unstable environments.

9.2.2.1.2 Performance Details

The following performance details provide information as to the degree of Lamarckian and Darwinian evolution being utilized in response to several different stability levels.

9.2.2.1.2.1 Stable Environment

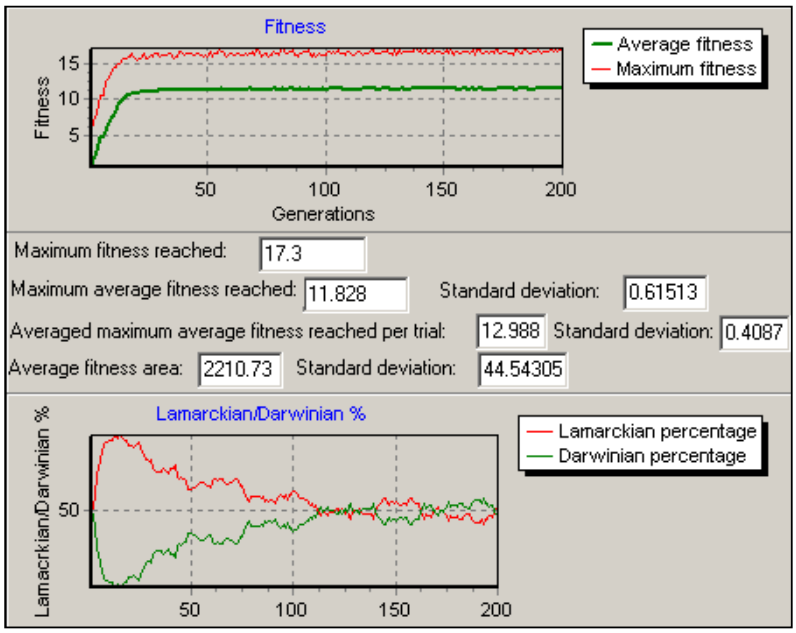


Figure 9-2 The heterogeneous population model's performance in the stable environment.

Figure 9-2 contains both the performance graph of our Darwinian/Lamarckian model's agents and the degree of Lamarckian or Darwinian evolution being utilized over the course of generations.

The above performance is clearly Lamarckian in nature, producing peak fitness levels within a few generations. This is explained by the bottom graph, indicating that Lamarckian agents come to quickly dominate the population within those first few generations. The Lamarckian agents' exceptional learning time, under stable conditions, is immediately selected for, changing the initial population consisting of equal numbers of Darwinian and Lamarckian agents to one that is dominated by the latter.

However, once peak fitness levels have been produced the population gradually progresses towards an equilibrium of both Darwinian and Lamarckian agents, indicating that a selective pressure towards Lamarckian agents no longer exists, since effective solutions have already been found and both Lamarckian and Darwinian agents are capable of maintaining these solutions. Thus, optimization is shown to

effectively adjust the nature of the agents' evolution according to the conditions of the environment and the stage of adaptation.

9.2.2.1.2.2 Fitness Consequence Alternations Every 25 Generations

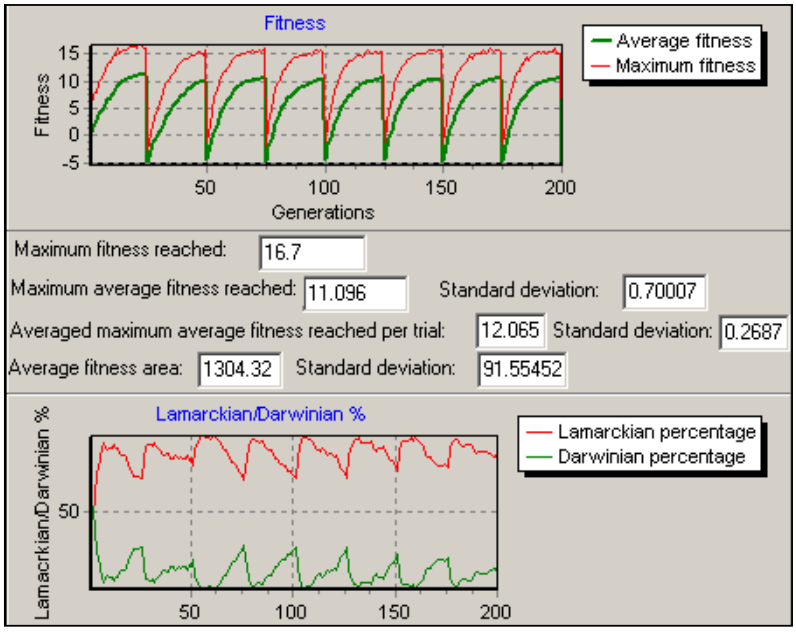


Figure 9-3 The heterogeneous population model's performance in an environment with fitness consequence alternations every 25 generations.

The above performance of Figure 9-3 is also typically Lamarckian. With reference to the bottom graph, it is quite clear that high levels of Lamarckian agents are maintained throughout the simulation. Under these conditions it seems that the increasing levels of instability require a more consistent use of Lamarckian evolution. This reinforces our previous conclusion that Lamarckian evolution does appear to maintain its effectiveness in response to a certain degree of increased instability. Note that the Lamarckian agents decrease in number as adaptation occurs with an immediate increase occurring after every fitness consequence reversal. This emphasizes both the effectiveness of Lamarckian evolution to deal with a change in conditions and the ability of our optimization technique to make use of the available options (Darwinian or Lamarckian agents) to appropriately deal react to the conditions of the environment throughout the course of the simulation.

This continues to be the case utilizing an environment with fitness consequence reversals every 7 generations.

9.2.2.1.2.3 Fitness Consequence Alternations Every 7 Generations

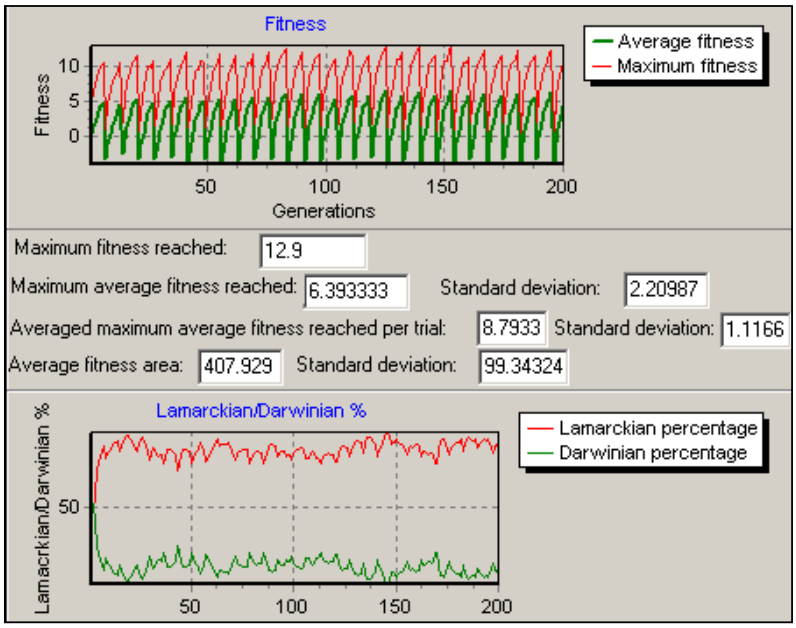


Figure 9-4 The heterogeneous population model's performance in an environment with fitness consequence alternations every 7 generations.

Once again Lamarckian agents are favoured, producing the highly oscillating fitness performance typical of the Lamarckian model. Recall that at this level of instability the Darwinian evolution model was beginning to show signs of stability and growth. However, even though the Lamarckian agents are prone to short periods of ineffectiveness, producing negative fitness scores, they are still favoured by selection over Darwinian agents. As a result, Lamarckian agents once again dominate the population over the course of the simulation. This is consistent with previous results indicating the superiority of the Lamarckian model at this level of stability.

The nature of our population changes quite rapidly, though, in response to further increases in instability.

9.2.2.1.2.4 Fitness Consequence Alternations After Every Generation

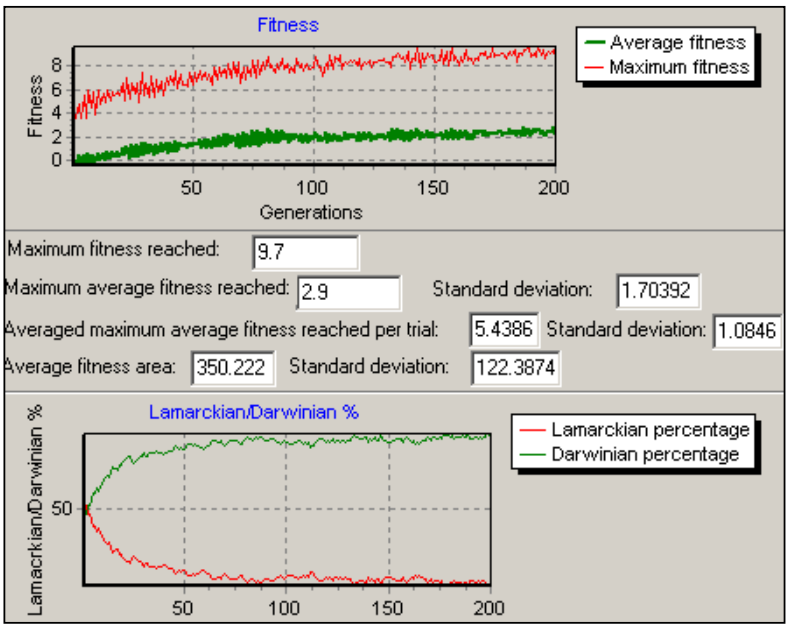


Figure 9-5 The heterogeneous population model's performance in an environment with fitness consequence alternations after every generation.

Figure 9-5 reveals that at the highest level of instability a Darwinian nature is clearly preferred. Agents utilizing population or generation level learning in conjunction with individual learning can appropriately adapt to such conditions and are, therefore, selected. As a result, Darwinian agents come to quickly dominate the population, producing solutions that demonstrate steadily increasing fitness levels under conditions of extreme instability.

Thus, in response to the highest levels of instability, the advantages of Darwinian evolution are selected to provide effective solutions not possible under a Lamarckian framework. In response to the stable and moderately unstable conditions the model utilizes evolution of a Lamarckian nature. Therefore, our hybrid Darwinian/Lamarckian heterogeneous population model demonstrates the ability to utilize either form of evolution as the situation demands.

9.3 Darwinian/Lamarckian Connection Model

Our second model utilizes neural network connections that can be of both a Lamarckian or Darwinian nature. A Lamarckian connection inherits the learnt changes to their ancestor connection's weight while a Darwinian connection inherits the innate (genotype) weight of their ancestor connection. Thus, the agents can posses neural networks that are both Lamarckian and Darwinian in nature.

Optimization is provided to modify the nature of the connections appropriate to the environment's characteristics.

9.3.1 Design

The requirements of our Darwinian/Lamarckian connection model are:

- Neural network connections that can be of either a Lamarckian or Darwinian nature.
- The ability to optimize such connections according to the requirements of the environment.

9.3.1.1 Darwinian/Lamarckian Neural Network Connections

Each connection is assigned an extra attribute (gene) determining its Lamarckian or Darwinian nature. At the end of each generation our model examines this attribute to determine whether an offspring connection was produced from a Lamarckian parent connection and should inherit the learnt knowledge of its ancestor connection or whether its parent connection was Darwinian and should inherit its ancestor connection's innate (genotype) knowledge. Thus, Darwinian or Lamarckian connections can be selected and reproduced according to the effectiveness of the role they play in the agents' neural networks adaptation to the environment. This, then, provides each agent with a neural network consisting of connections that can be of either a Lamarckian or Darwinian nature.

9.3.1.2 Optimization

Each connection can have its Darwinian or Lamarckian nature mutated to become its converse. As before, agents are chosen for mutation with a probability determined by the population mutation rate with each neural network connection then being chosen with a probability determined by the individual mutation rate. The chosen connections are mutated to become their converse. Thus, various configurations of Lamarckian and Darwinian connections can be evaluated (via selection) and reproduced according to their success in the environment.

9.3.1.3 Simulation Framework and Parameters

Our hybrid Darwinian/Lamarckian connection model is applied utilizing the same simulation framework as our previous models.

The important constants are:

- A generation lifetime of 500
- A simulation length of 200 generations
- A population of 25
- 1 food object in the world at any time
- 1 poison object in the world at any time
- 30 trials per sample

The following evaluation utilizes connections that are randomly initialized as either Darwinian or Lamarckian. Thus, the agents' neural networks are not initially biased towards either a Lamarckian or Darwinian nature.

The previous reinforcement feedback values of 3, -3 and –0.019 are utilized for locating food, poison and per action, respectively, since these have been found to be highly successful in the initially stable environment, complementing an effective Lamarckian nature. The previous results demonstrate that these values are also highly compatible with agents demonstrating a Darwinian nature in unstable environments. Thus, it is the task of this model to utilize these values to provide an effective performance in the environment, from stable to unstable conditions.

As before, we also performed experiments to evaluate the potential benefits of allowing these values to be adjusted, via optimization. However, fixed reinforcements were found to provide the most effective results in conjunction with an individual mutation rate that is allowed to self-optimize. Refer to appendix O for complete data concerning the other options.

9.3.2 Evaluation of Our Darwinian/Lamarckian Connection Model

9.3.2.1 Performance in Stable to Unstable Environments

9.3.2.1.1 Overview

The following provides an evaluation of our Darwinian/Lamarckian connection model utilizing fixed reinforcements of 3, -3 and –0.019 for locating food, poison and per action, respectively, with a population mutation rate of 0.8 and an individual mutation rate that is allowed to optimize. Refer to appendix O for complete data regarding this evaluation and the model's performance utilizing the other experimental options.

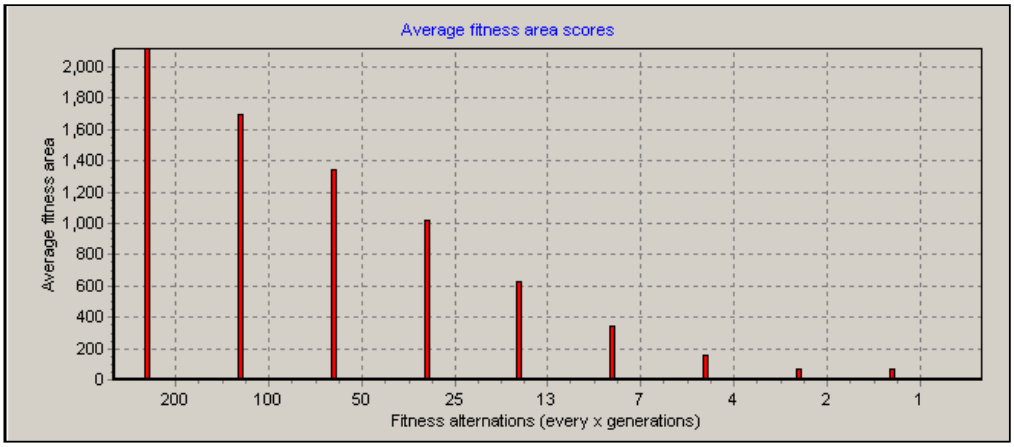


Figure 9-6 Average fitness area scores produced by our hybrid Darwinian/Lamarckian connection model in stable to highly unstable environments.

Our hybrid Darwinian/Lamarckian connection model provides a reasonably adequate performance in response to the environment's conditions, from stable to unstable (Figure 9-6). Although it is clearly less effective than the previous model, it does demonstrate a certain level of adaptability to the most unstable environments. In this regard it is superior to our evolution, reinforcement learning and Lamarckian models. However, it does appear that utilizing connections that are either Lamarckian or Darwinian does not provide a highly effective degree of adaptation to our examined problem. Although solutions of an equivalent nature to those of the previous hybrid model and the Darwinian model are possible, extending the solution space by requiring that each connection be correctly configured (Lamarckian or Darwinian) may be responsible for hindering the search processes ability.

Another disadvantage of our Darwinian/Lamarckian connection model is the brittle nature of each connection's degree of inheritance. That is, they are either Lamarckian or Darwinian. Thus, mutating a connection's inheritance nature can radically alter its functioning in relation to the solution being applied.

This does not mean to say that such models do not posses potential advantages, utilizing neural networks consisting of both Lamarckian and Darwinian components

may be a useful approach to solving certain kinds of problems. However, in response to the examined problem this model appears to be of little benefit.

A potential means of improving the model may be to incorporate connections that can utilize degrees of inheritance, from 100% Darwinian to 100% Lamarckian, thereby solving this model's brittle nature. Our model implementing such a technique is discussed in section 9.5. The next section, though, discusses our work utilizing neural networks that are capable of inheriting the learnt knowledge of their parents according to various degrees.

9.4 Global Degree of Neural Network Inheritance Model

Our third Lamarckian/Darwinian model utilizes connections that inherit the learnt knowledge of their ancestors according to a certain, globally specified degree. Thus, the connections are not merely Darwinian or Lamarckian, but can also demonstrate the ability to partially inherit the learnt changes of their parent connections.

The degree of inheritance is global in the sense that all connections in an agent's neural network inherit the learnt knowledge of their ancestor connections to an equivalent extent, determined by a single inheritance parameter. This inheritance parameter is included as a gene in our agents' description and is, thus, open to optimization providing our agents with the ability to adjust their degree of inheritance appropriate to the nature of the environment.

9.4.1 Design

The requirements of our global degree of neural network inheritance model are:

- Neural network connections that posses the ability to inherit learnt knowledge according to a degree determined by a global inheritance parameter.
- The ability to optimize this parameter to provide the appropriate degree of inheritance in response to the nature of the environment.

9.4.1.1 Partially Darwinian/Lamarckian Neural Network Connections

Our neural network connections inherit the learnt knowledge of their ancestors according to the following function:

$$W_o = W_i + I_g (W_l - W_i)$$

where W_o is the weight of the offspring connection, W_i is the innate weight of the parent connection, W_l is the learnt weight of the parent connection and I_g is the neural network's global degree of inheritance parameter.

Thus, offspring connections inherit the learnt changes of their parent connections to an extent determined by a global degree of inheritance parameter associated with each neural network.

9.4.1.2 Optimization

To provide optimization this parameter is included as a gene in our agents' description. Thus, it is available to the processes of selection and mutation.

As before, agents are chosen for mutation with a probability depending on the population mutation rate. Should an agent be chosen then its global degree of inheritance gene is mutated according to the following:

$$\Delta\text{global degree of inheritance parameter} = \begin{cases} x & \text{if a random digit is 0} \\ 0 & \text{if the random digit is 1} \\ -x & \text{if the random digit is 2} \end{cases}$$

where x is a small, constant value.

Our global degree of inheritance parameter is allowed to move in the range 0…1, with a degree of change (x) of 0.1 to provide adequate movement and a sufficient range of inheritance. Thus, our agents can be anything from 100% Darwinian to 100% Lamarckian, with the appropriate degree of inheritance being determined by the selective pressure of the environment.

9.4.1.3 Simulation Framework and Parameters

Our hybrid Darwinian/Lamarckian global degree of neural network inheritance model is applied utilizing the same simulation framework as our previous models.

The important constants are:

- A generation lifetime of 500
- A simulation length of 200 generations
- A population of 25
- 1 food object in the world at any time
- 1 poison object in the world at any time
- 30 trials per sample

The following evaluation utilizes agents that posses a global degree of inheritance parameter initialized to 0.5. Thus, their neural networks are initialized to inherit half the learnt changes of their parents' neural networks. This provides a useful midpoint value that can be further optimized according to the particular conditions of the environment.

Once again, the previous reinforcement feedback values of 3, -3 and –0.019 are utilized for locating food, poison and per action, respectively, since these have been found to complement the effectiveness of Lamarckian evolution in the initially stable environment. It is up to our model to select its Lamarckian nature in order to make effective use of these parameters in the initially stable environment and to demonstrate its adaptive capabilities in response to increasing levels of instability.

As before, we also performed experiments to evaluate the potential benefits of allowing these parameters to be adjusted, via optimization, should changes in stability require different values. The most effective performance, though, was produced utilizing the fixed reinforcement feedback values of 3, -3 and –0.019, for locating food, poison and per action, respectively, with a population mutation rate of 0.8 and a fixed individual mutation rate of 0.1. This performance is presented in the following evaluation. Refer to appendix P for complete data regarding this evaluation and the other options.

9.4.2 Evaluation of Our Global Degree of Neural Network Inheritance Model

9.4.2.1 Performance in Stable to Unstable Environments

9.4.2.1.1 Overview

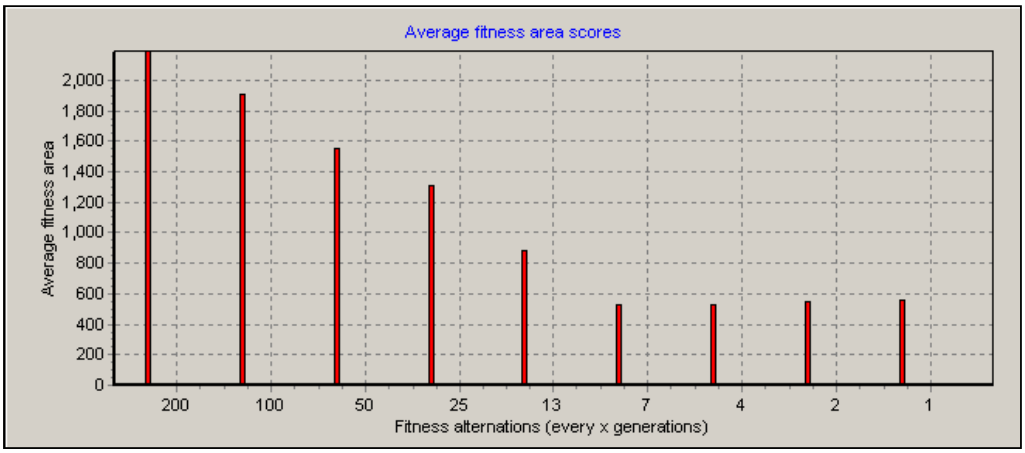


Figure 9-7 Average fitness area scores produced by our hybrid Darwinian/Lamarckian global degree of neural network inheritance model in stable to highly unstable environments.

Figure 9-7 demonstrates that our Darwinian/Lamarckian global degree of neural network inheritance model is also highly effective in dealing with increasing levels of instability. In fact it can be considered the most effective of our Darwinian/Lamarckian hybrid models, producing average fitness area levels above 400 in the four most unstable environments. Note that these levels are maintained with no further degradation of performance in response to the highest levels of instability. Thus, demonstrating this model's ability to assimilate increasing instability, thereby maintaining its high performance levels.

Clearly, then, this model effectively utilizes the abilities of both Darwinian and Lamarckian evolution, enhancing its ability to adapt to the environment's conditions from stable to unstable.

The following provides details regarding this model's performance in response to several degrees of stability.

9.4.2.1.2 Performance Details

9.4.2.1.2.1 Stable Environment

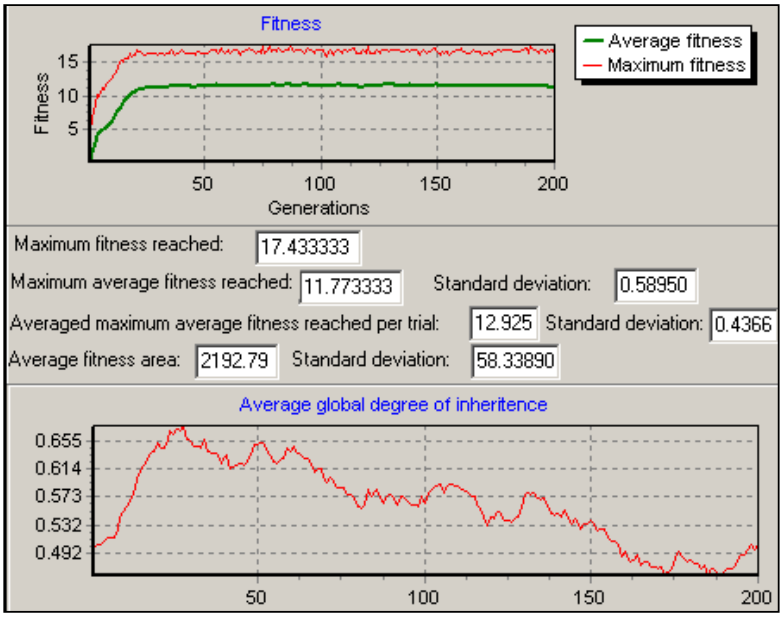


Figure 9-8 The global degree of neural network inheritance model's performance in the stable environment.

Our global connection degree of inheritance model demonstrates a typically Lamarckian performance in the stable environment, quickly reaching peak fitness levels within a few generations (Figure 9-8). Also shown in the figure above is a graph of the average global degree of inheritance occurring in the population. This is an average measure of the degree to which the agents' neural networks are inheriting the learnt knowledge of their parents. Like each agent's neural network, the average degree of inheritance occurring can range between 0 (100% Darwinian) and 1 (100% Lamarckian). Note that this measure quickly rises within a few generations from its initial value of 0.5 to reach a maximum level for the simulation, after which it once again decreases towards the midpoint value. This reflects the initial degree of Lamarckian evolution required to produce the peak performance levels. Once effective solutions have been produced it appears that the agents high degree of Lamarckian inheritance is no longer needed. Instead the decreasing average global degree of inheritance indicates that agents are being selected for a lower degree of inheritance, finally settling towards a 50% Lamarckian/Darwinian nature. Thus, even though the environment is stable it appears that agents demonstrating a 100% Lamarckian nature will not necessarily dominate throughout the simulation, although a high degree of Lamarckian inheritance is clearly needed in the earlier stages of adaptation.

The following reveals how this adaptive ability is utilized under increasing levels of instability.

9.4.2.1.2.2 Fitness Consequence Alternations Every 25 Generations

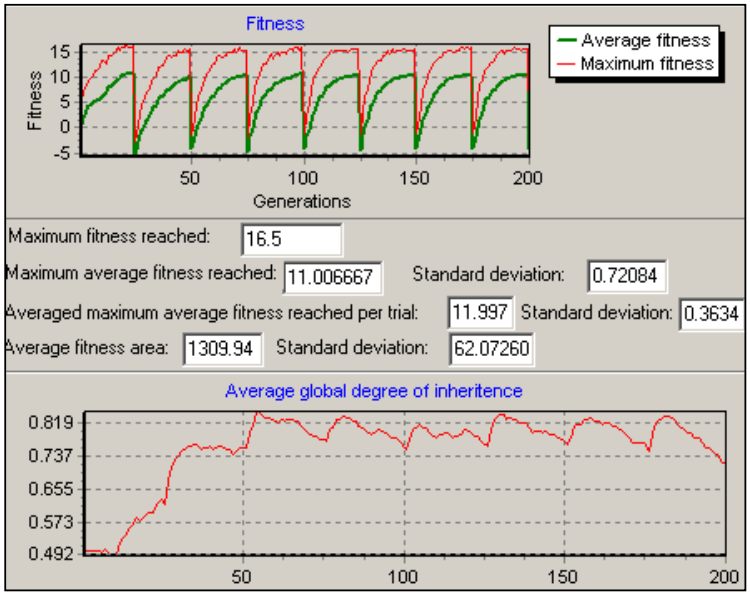


Figure 9-9 The global degree of neural network inheritance model's performance in an environment with fitness consequence alternations every 25 generations.

Our model's performance in response to fitness consequence reversals every 25 generations is also of a Lamarckian nature (Figure 9-9). After 50 generations the agents' average global degree of inheritance has risen to a maximum level indicating a high degree of Lamarckian inheritance. This level is then maintained throughout the rest of the simulation in response to the continuing fitness consequence reversals. Thus, an essentially Lamarckian nature is utilized throughout the simulation in order to effectively adapt to this level of instability. This is consistent with our previous results indicating the superiority of Lamarckian evolution in the more stable environments.

The situation does change, though, in response to the more unstable environments.

9.4.2.1.2.3 Fitness Consequence alternations Every 7 generations

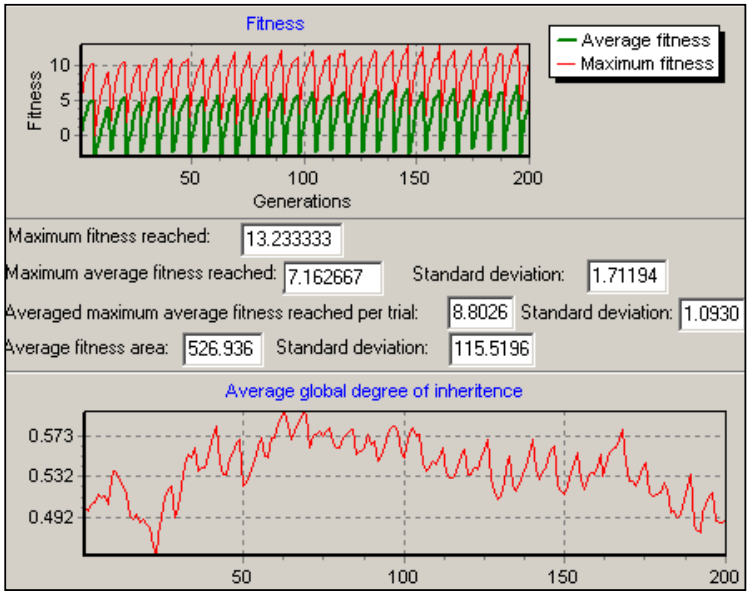


Figure 9-10 The global degree of neural network inheritance model's performance in an environment with fitness consequence alternations every 7 generations.

The data above (Figure 9-10) indicates that our model initially prefers agents demonstrating a higher degree of Lamarckian inheritance than the starting level of 0.5. However, the highest levels of Lamarckian inheritance are not maintained and slowly decline over the course of the simulation, ultimately producing an average global degree of inheritance of around 0.5. This is in contrast to our heterogeneous population model that maintained high levels of Lamarckian agents throughout the simulation in response to the same level of instability. It appears, then, that this model is making use of its ability to produce agents that utilize a partial degree of inheritance in response to the conditions of this environment, producing an average fitness area of 526.936 as compared to the heterogeneous population model's average fitness area of 407.929.

Recall that our heterogeneous population model utilizes agents that are either Darwinian or Lamarckian and, thus, cannot produce agents that make use of partial inheritance. In contrast our global connection degree of inheritance model can, allowing it to produce solutions that exist in the range between 100% Darwinism and 100% Lamarckism. As the above results indicate this can prove to be a useful ability.

It appears to also have advantages in response to the most unstable environment.

9.4.2.1.2.4 Fitness Consequence Reversals After Every Generation

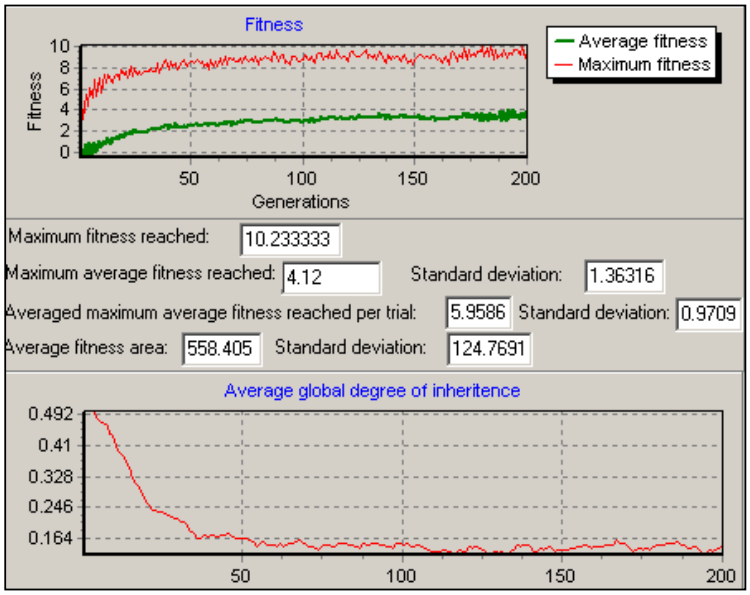


Figure 9-11 The global degree of neural network inheritance model's performance in an environment with fitness consequence alternations after every generation.

The above figure (Figure 9-11) presents our global connection degree of inheritance model's performance in the most unstable environment. This is the most effective performance of all the models, producing a stable and constant growth in fitness levels.

Note that the average global degree of inheritance measure declines quite rapidly to reflect the increasingly Darwinian nature of the population's agents and these levels are maintained throughout the rest of the simulation. Recall that the heterogeneous population model's performance reflected a similar nature, quickly producing a population dominated by Darwinian agents. However, our global connection degree of inheritance model does appear to be superior, producing more effective fitness measures. It appears that the capability of the agents to gradually alter their degree of inheritance benefits their evolution towards more effective solutions. As a result, this model demonstrates the ability to effectively move from its initial midpoint inheritance value of 0.5 to a Darwinian nature of inheritance in order to adapt to the most unstable environment.

In conclusion then, this model demonstrates the ability to utilize the appropriate degree of inheritance according to the requirements of the environment, gradually optimizing the degree of inheritance required in the range between a 100% Lamarckian nature to a 100% Darwinian nature. Thus, the model can effectively adapt to changes in stability from the most stable environments to the most unstable.

9.5 Local Degree of Connection Inheritance Model

Our last Darwinian/Lamarckian model utilizes connections that inherit the learnt knowledge of their ancestors according to their own individual degree. Thus, the connections are of a similar nature to the previous model, demonstrating the ability to partially inherit the previously learnt knowledge of their parents. However, the degree to which they inherit learnt knowledge is unique to each connection. This is achieved by assigning each connection a local degree of inheritance parameter.

These parameters are included as genes in the agents' description and are, thus, also open to the processes of selection and mutation, providing the model with the ability to optimize each connection's degree of inheritance according to the requirements of the environment. Thus, our agents posses neural networks consisting of connections that are capable of inheriting the learnt knowledge of their parent connections to differing degrees, as determined by an inheritance parameter local to each connection.

9.5.1 Design

The requirements of our local connection degree of inheritance model are:

- Neural network connections that posses the ability to inherit learnt knowledge according to a degree determined by a local inheritance parameter.
- The ability to optimize this parameter to provide the appropriate degree of inheritance per connection in response to the nature of the environment.

9.5.1.1 Local Degree of Inheritance per Connection

Our connections are of the same nature of the previous model except that they now contain their degree of inheritance as a local parameter. Thus, their inheritance function becomes:

$$W_o = W_i + I_l (W_l - W_i)$$

where W_o is the weight of the offspring connection, W_i is the innate weight of the parent connection, W_l is the learnt weight of the parent connection and I_l is the local degree of inheritance parameter.

9.5.1.2 Optimization

Since each connection now contains a gene describing the degree of its inheritance, this aspect of its functioning is available to selection and mutation, thereby providing the capability for optimization.

Should an agent be chosen for mutation then its neural network connections' degree of inheritance parameters may be selected for mutation according to a probability determined by the individual mutation rate.

Each connection's local degree of inheritance is then mutated according to:

$$\Delta\text{local degree of inheritance parameter} = \begin{cases} x & \text{if a random digit is 0} \\ 0 & \text{if the random digit is 1} \\ -x & \text{if the random digit is 2} \end{cases}$$

where x is a small, constant value.

Our local degree of inheritance parameters are allowed to move in the range 0…1, with a degree of change (x) of 0.1 to provide adequate movement and a sufficient range of inheritance. This allows our agents' connections to be anything from 100% Darwinian to 100% Lamarckian, with the appropriate degree of inheritance being determined by the selective pressure of the environment in response to the role each connection plays in the agents' neural networks.

9.5.1.3 Simulation Framework and Parameters

Our hybrid Darwinian/Lamarckian local degree of connection inheritance model is applied utilizing the same simulation framework as our previous models.

The important constants are:

- A generation lifetime of 500
- A simulation length of 200 generations
- A population of 25
- 1 food object in the world at any time
- 1 poison object in the world at any time
- 30 trials per sample

The following evaluation utilizes agents that posses local degree of inheritance parameters initialized to 0.5 per connection. Thus, the agents' neural network connections are initialized to inherit half the learnt changes of their parent neural network connections. This provides a useful midpoint value that can be further optimized according to the nature of the environment and the role each connection plays in producing an agent's response to that environment.

As before, reinforcement feedback values of 3, -3 and –0.019 are utilized for locating food, poison and per action, respectively, to provide the appropriate feedback for an effective solution to the initially stable environment. As previously mentioned, they have also been found to be compatible with solutions to the unstable environment. Thus, it is up to our local connection degree of inheritance model to utilize this feedback in order to produce effective solutions to the initially stable environment and in response to increasing levels of instability.

We also performed experiments to evaluate the potential benefits of allowing these parameters to be adjusted, via optimization, should changes in stability require different values. The most effective performance, though, was produced using the fixed reinforcements of 3, -3 and –0.019 for locating food, poison and per action respectively, with a population mutation rate of 0.8 and a fixed individual mutation rate of 0.1. Average fitness area results provided by this performance are presented in

the following evaluation. Refer to appendix Q for complete data regarding this performance and the other experimental options.

9.5.2 Evaluation of Our Local Degree of Connection Inheritance Model

9.5.2.1 Performance in Stable to Unstable Environments

9.5.2.1.1 Overview

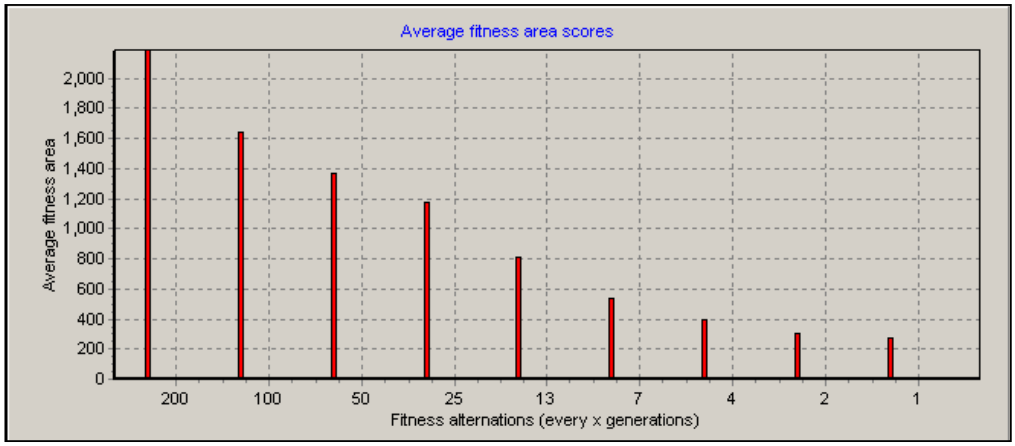


Figure 9-12 Average fitness area scores produced by our hybrid Darwinian/Lamarckian local degree of connection inheritance model in stable to highly unstable environments.

Although not as effective as the previously presented model utilizing a global degree of inheritance, our local degree of connection inheritance model demonstrates an effective ability to adapt to the most stable environment and the increasing levels of instability (Figure 9-12). It appears that this model's ability to utilize connections that are capable of partial degrees of inheritance allow it to overcome the brittleness of the Darwinian/Lamarckian connection model, capable of only utilizing connections that are either Darwinian or Lamarckian. Thus, even though the search space has been extended, this model is clearly capable of adjusting each connections degree of inheritance to provide effective solutions in response to the range of environments, from stable to unstable. Although not quite as effective as the heterogeneous population model and the global degree of connection inheritance model, this model performs effectively in response to the examined problem and provides the potential to produce effective neural networks consisting of both Lamarckian and Darwinian aspects that may prove useful to solving problems of a different nature.

9.6 Adaptive Behaviour and Neural Network Knowledge of Our Darwinian/Lamarckian Hybrids

Our hybrid models utilize agents that are of a Darwinian or Lamarckian nature according to the requirements of the situation. Two of these models utilize agent neural networks that are capable of containing both Darwinian and Lamarckian connections. That is, individual connections can be configured to be of a Darwinian or Lamarckian nature. Although this approach has the potential to provide additional adaptive qualities, it achieved only a moderate degree of success relative to the heterogeneous population model and the global degree of connection inheritance model. It appears that the potential for producing adaptive behaviour and neural networks of a different quality is not required in response to the characteristics of our simulations.

Our heterogeneous population model utilizes agents of either a Darwinian or Lamarckian nature according to the demands of the environment. Under reasonably stable conditions, agents are selected that demonstrate Lamarckian adaptive characteristics. In response to the most unstable environments, agents of a Darwinian nature are preferred. Thus, the agents demonstrate behaviour and neural network knowledge that is either of a Lamarckian or Darwinian nature depending on the characteristics of the environment.

Our global degree of neural network inheritance model appears to be the most successful of the hybrids, providing the greatest degree of adaptability in the most unstable environments and the high fitness measures typical of Lamarckian evolution in the stable environments.

This model also creates agents that are essentially of a Lamarckian or Darwinian nature in response to the examined environments. However, it does demonstrate the ability to utilize partial degrees of inheritance in response to environments of a certain stability level. Recall the average global degree of inheritance utilized in an environment with fitness consequence alternations every 7 generations (Figure 9-10). There the degree of inheritance is maintained near the midpoint value of 0.5, indicating that agents are being created to utilize a partial degree of inheritance in order to effectively adapt to the environment.

In addition, this model's ability to utilize a degree of inheritance appears to benefit the search's ability to produce effective neural network knowledge. With reference to the model's performance in the most unstable environment (Figure 9-11), the agents appear to be more effectively moved towards the appropriate level of Darwinian inheritance as opposed to the heterogeneous population model's agents that can only ever be of a either a Darwinian or Lamarckian nature (Figure 9-5). This benefits the development of effective neural network knowledge as the solutions can be more gradually moved towards the appropriate level of inheritance, thereby making it easier for neural network knowledge to adapt to a different degree of inheritance as compared to an either/or Darwinian/Lamarckian nature. As a result, the development of neural network knowledge can form a more complementary relationship with a changing degree of inheritance.

Thus, our global degree of neural network inheritance model benefits the development of effective neural network knowledge by choosing the appropriate degree of inheritance required in response to the stage of adaptation and the conditions of the environment.

9.7 Conclusion

Our hybrid Darwinian/Lamarckian models demonstrate a superior adaptability, appropriately adjusting the required degree of inheritance according to the conditions of the environment.

The heterogeneous population model is shown to be highly effective in adjusting its population's agents to utilize the appropriate form of inheritance, from Lamarckian in stable conditions to Darwinian in unstable conditions. Our results demonstrate that the model is capable of optimizing the percentage of either type of agent according to the characteristics of the environment and the stage of the simulation. It is shown to effectively combine the advantages of Darwinian and Lamarckian evolution models to produce a more effective hybrid capable of adapting to the entire range of our examined environment's conditions, from stable to unstable.

The Darwinian/Lamarckian connection model is shown to be the least effective of our hybrid models. Although demonstrating an adequate level of adaptability in response to the examined environmental conditions, it can not match the performance levels achieved by the pure Darwinian and pure Lamarckian models in their favoured conditions. It appears that the search's larger solution space, due to the model's requirement that each connection be appropriately configured (Lamarckian or Darwinian), and the brittle nature of this model hinder its ability to find effective solutions to the examined problem.

The global degree of neural network inheritance model provides the most successful performance, exceeding that of the heterogeneous population model in the most unstable conditions. Utilizing agents capable of various degrees of inheritance allows the model to more effectively move its search towards the appropriate solutions required in response to the particular conditions of an environment. As a result, the appropriate degree of inheritance can be adjusted and utilized according to the stage of adaptation and the environment's level of stability.

The local degree of connection inheritance model is also shown to be effective in adapting to the environment. Although each connection's degree of inheritance must be appropriately configured, thereby extending the solution space, effective agents are produced in the entire range of environmental conditions.

Thus, all our hybrid Darwinian/Lamarckian models demonstrate an effective ability to adapt to the entire range of examined environmental conditions, with the heterogeneous population model and the global degree of neural network inheritance model providing the most successful results. By combining the advantageous of both Darwinian and Lamarckian evolution we have, thus, provided several models capable of adapting to a range of environment conditions, from stable to unstable,

appropriately adjusting the required degree of inheritance according to the nature of the environment.

We have also validated the results of previous work, indicating the nature of Lamarckism to favour stable environments and the nature of Darwinism to favour unstable environments [Sasaki and Tokoro, 1997; Sasaki and Tokoro, 1999]. In addition we have provided and evaluated the use of inheritance optimization mechanisms resulting in the creation of hybrid Darwinian/Lamarckian models capable of effectively adjusting and utilizing the appropriate degree of inheritance according to the characteristics of the examined environment, ranging from stable to unstable conditions.

The following chapter provides a final comparison of the results produced by our most effective hybrid Darwinian/Lamarckian model with those of the previously presented evolution and reinforcement learning models.

Chapter 10: Comparison

10.1 Introduction

The following provides a comparison of the best performances of all our evolution and reinforcement learning models in response to the range of examined environments. The models are compared with regard to their overall performance in the entire range of environmental conditions, from stable to unstable, and, in addition, a more detailed comparison is also provided with regard to their performance in response to several particular degrees of environmental stability.

10.2 Performance Overview

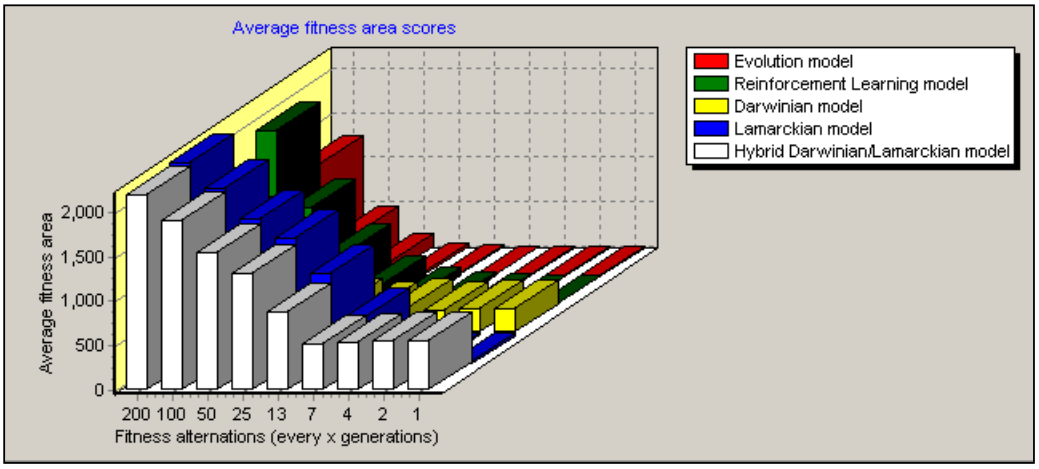


Figure 10-1 3d graph demonstrating the performance of our evolution and reinforcement learning models in the environments, from stable to unstable.

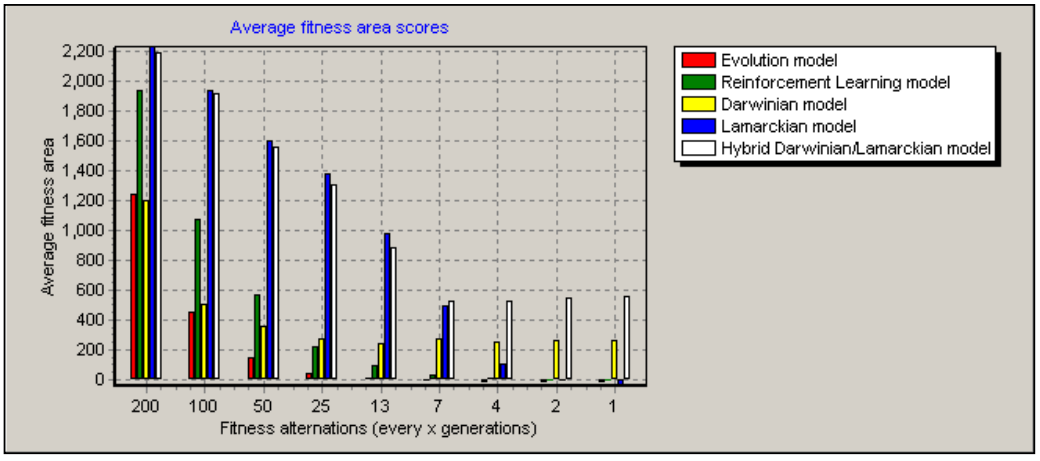


Figure 10-2 2d graph demonstrating the performance of our evolution and reinforcement learning models in the environments, from stable to unstable.

Figure 10-1 and Figure 10-2 represent 3 dimensional and 2 dimensional perspectives, respectively, of all our evolution and reinforcement learning models' performances in the environments from stable to unstable. They were produced utilizing average fitness area data, the most useful measure in terms of the models overall performance. Refer to appendix Q for complete data regarding the other performance measures.

The evolution model is clearly the least effective, followed by the reinforcement learning model. Our Darwinian evolution and reinforcement learning model, equivalent to the evolution model in the stable environments, demonstrates the ability to adapt far more effectively in response to increasing levels of instability. In comparison to the evolution model, reinforcement learning model and the Lamarckian model, it appears ideally suited to highly unstable environments.

However, in response to the stable environments the Lamarckian model appears to be the most effective, only degrading in response to the most unstable environments, thus, reinforcing our previous statements regarding the nature of Lamarckian evolution to be ideally suited towards stable environments and Darwinian evolution's superior adaptive nature towards more unstable environments.

Our hybrid Darwinian/Lamarckian model demonstrates the ability to effectively adapt to the entire range of environments, from stable to unstable. The advantages of both Lamarckian and Darwinian evolution are combined in the form of a single model, with the required degree of inheritance being adjusted in response to the conditions of the environment. Thus, an effective Lamarckian performance is produced in the stable environments with a performance of a Darwinian nature being provided in response to the more unstable environments. These results, then, clearly demonstrate the advantages of hybrid models utilizing both Lamarckism and Darwinism in response to environments demonstrating changing levels of stability.

The following provides details regarding the performance of the models' in response to several particular degrees of environmental stability.

10.3 Performance Details

10.3.1 Stable Environment

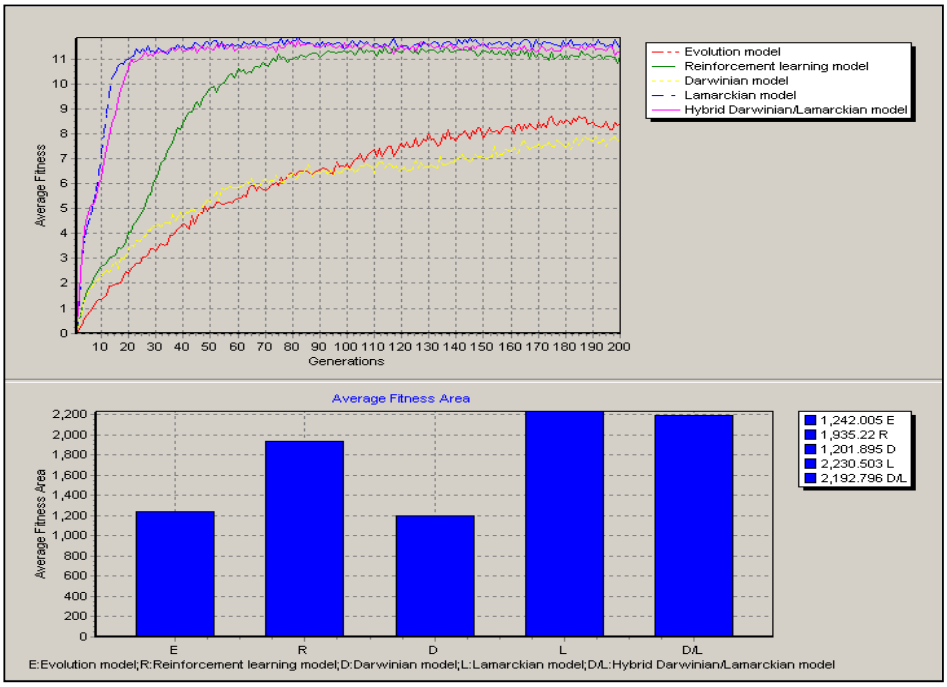


Figure 10-3 The performance of our evolution and reinforcement learning models in the stable environment.

With reference to the figure above (Figure 10-3) our evolution model's performance (red graph) demonstrates that it has the ability to steadily produce increasingly effective agents over the course of 200 generations. However, in comparison to most of the other models, this performance is both slower and appears to climax at lower peak fitness values.

The Darwinian evolution model, utilizing lifetime reinforcement learning without inheritance, does not perform any better. Although superior in the initial stages of the simulation, its performance ultimately degrades to become slightly worse than that of the evolution model. However, recall that this performance of the Darwinian evolution model utilized the option of reinforcement feedback optimization, necessary to produce an effective performance in the unstable environments requiring a higher degree of learning. In the stable environment this option is slightly detrimental to the models performance, increasing the search's solution space and, thus, slightly hindering its ability to produce effective agents in response to these conditions – fixed reinforcements capable of producing a slightly better performance than that of the evolution model, but ultimately ineffective in the more unstable environments.

Our reinforcement learning model is clearly far more effective in adapting to the stable environment. Our agents utilize their ability to learn throughout the course of the simulation, quickly assimilating knowledge regarding their environment and, as a result, producing peak fitness levels within 100 generations.

Combining reinforcement learning and evolution in a Lamarckian framework, thereby utilizing the processes of selection and reproduction to amplify correctly learnt knowledge, produces an even more effective performance. Within 50 generations peak fitness levels are produced that exceed those of the reinforcement learning model's throughout the entire course of its simulation. Clearly, selection and reproduction of inherited knowledge can be highly effective in stable environments.

Our hybrid Darwinian/Lamarckian model performs at an equivalent level of effectiveness, utilizing its ability to adjust the agents' degree of inheritance according to the conditions of the environment. As was shown in the previous chapter, the agents' degree of inheritance is quickly and effectively moved towards a Lamarckian nature in response to the environments stable conditions, therefore demonstrating its ability to produce effective agents of the appropriate nature in response to the demands of the environment.

The following demonstrates that this is also true in response to increasing levels of instability.

10.3.2 Fitness Consequence Alternations Every 25 generations

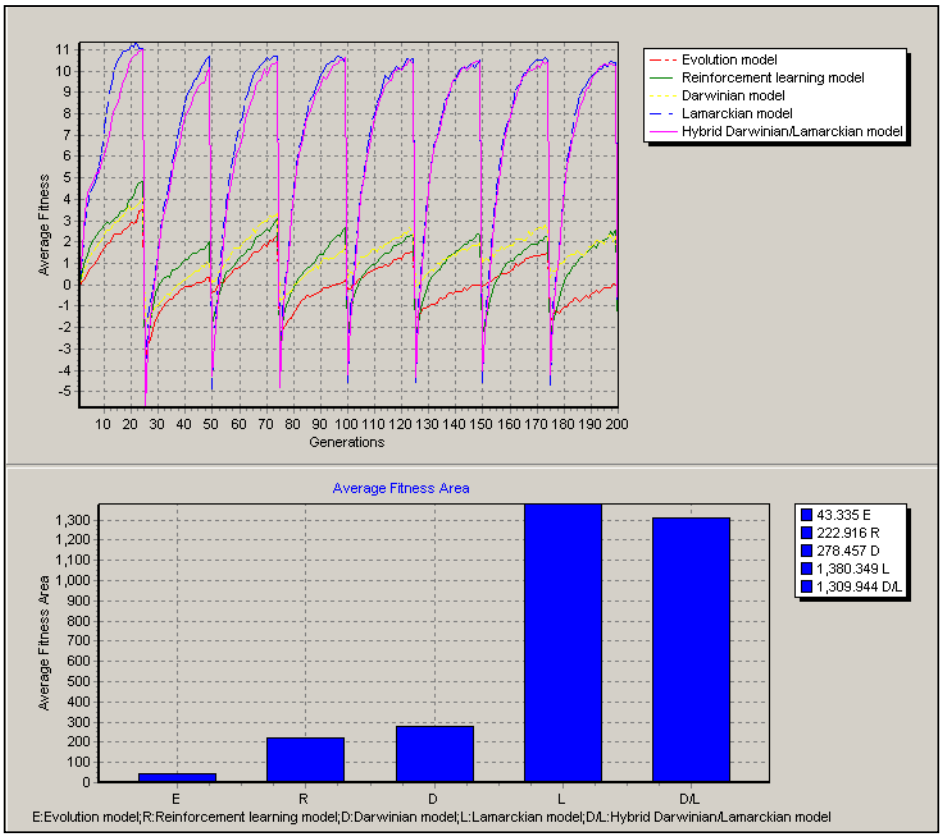


Figure 10-4 The performance of our evolution and reinforcement learning models in an environment with fitness consequence reversals every 25 generations.

Figure 10-4 demonstrates how our models react to fitness consequence reversals every 25 generations.

The performance of our evolution model is clearly the worst of the lot, being highly disrupted by the fitness consequence changes. The nature of its agents to contain fixed individual knowledge, modified over generations, results in slow unlearning/relearning times and, thus, an ineffective ability to adjust to the changes.

The reinforcement learning model provides the agents with the ability to learn within their individual lifetimes, thereby producing agents that are capable of a greater ability to unlearn incorrect knowledge and, thus, reducing the impact of the fitness consequence reversals. However, as a result of this learning nature, it is also required that appropriate behaviour, previously incorrect and unlearnt, be relearnt, since the agents are incapable of remembering conflicting solutions across the generations. This, though, is not the case with our Darwinian evolution and reinforcement learning model.

The Darwinian model demonstrates an ability to utilize the agents' innate, genotype knowledge to effectively reduce the impact of the fitness consequence reversals. As chapter 7 demonstrated, this knowledge is initially evolved to follow learning and directly discriminate between the food and poison objects of a particular period. As a result, it is also highly disrupted by the fitness consequence reversals – producing agents with incorrect, innate knowledge that must be unlearnt. However, after a number of generations the agents' genotype knowledge ultimately demonstrates a tendency to maintain an effectiveness evaluation near 0. Since the agents start their lives utilizing this knowledge and, thus, with an equivalent initial effectiveness level to be improved by learning, an effective means is created to buffer against the negative impact of the fitness consequence reversals. As chapter 7 later revealed, the agents' innate knowledge is not merely used as a buffer, but in fact comes to complement learning in order to improve the agents ability to assimilate change. However, at this level of stability the Darwinian model's effectiveness is essentially equivalent to that of the reinforcement learning model.

The last two models, though, demonstrate a much greater level of effectiveness. Both the Lamarckian model and our hybrid Darwinian/Lamarckian model demonstrate the ability to rapidly unlearn incorrect solutions and relearn appropriate behaviour after each fitness consequence reversal. Although their performance is disrupted by the fitness consequence reversals, both models make use of selection and reproduction of learnt knowledge to provide an effective learning ability capable of quickly adjusting to the new conditions.

10.3.3 Fitness Consequence Alternations Every 7 generations

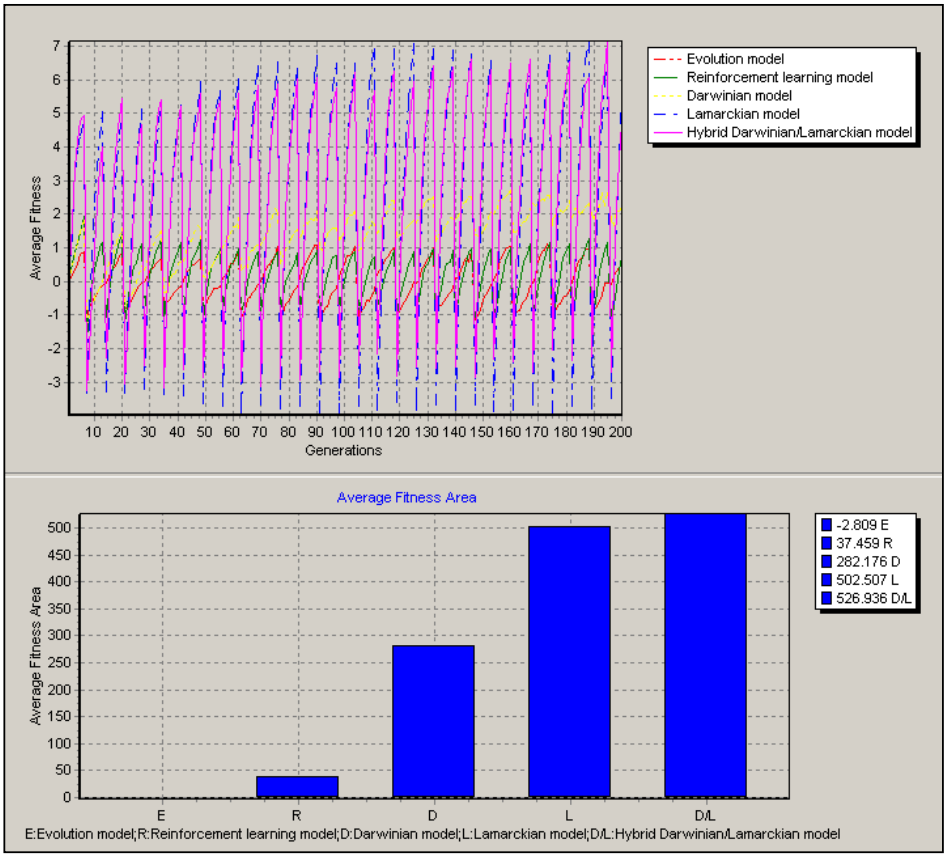


Figure 10-5 The performance of our evolution and reinforcement learning models in an environment with fitness consequence reversals every 7 generations.

In response to an increase in the environments instability, utilizing fitness consequence reversals after every 7 generations, most of the models react with a performance that is wildly oscillating in nature (Figure 10-5).

Both the evolution and reinforcement learning models are incapable of producing any level of effectiveness under these conditions, note the negative average fitness area of the evolution model and the low average fitness area of the reinforcement learning model.

The interesting feature of these performances is the nature of the Darwinian model to produce steadily increasing fitness levels without the oscillating nature of the other models. Clearly, at this level of instability our Darwinian model's increasing exposure to the environments changes is resulting in the assimilation of them, thereby providing an improving, stable performance over the course of generations. However, under these conditions a Lamarckian nature still produces the highest average fitness area scores despite a highly oscillating nature. Note the equivalent performance of our Lamarckian model and our hybrid Darwinian/Lamarckian model, indicating the preference of the latter towards a high degree of Lamarckian inheritance.

The situation changes quite dramatically, though, in response to increasing levels of instability.

10.3.4 Fitness Consequence Reversals After Every Generation

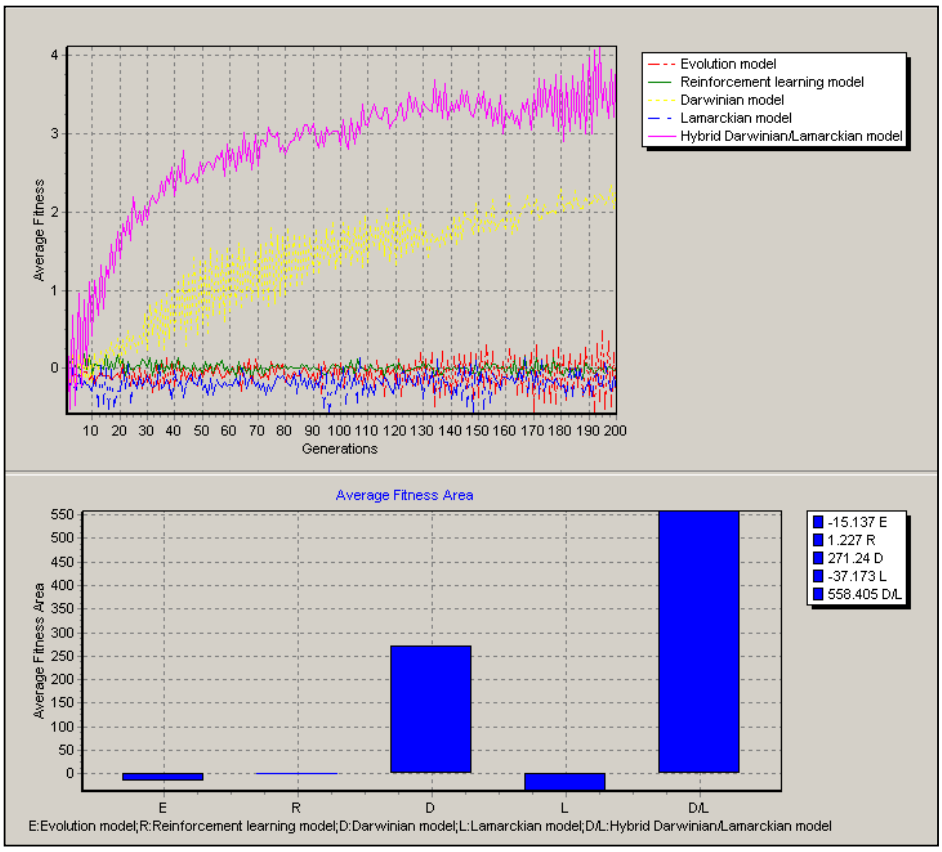


Figure 10-6 The performance of our evolution and reinforcement learning models in the most unstable environment, utilizing fitness consequence reversals after every generation.

Only our Darwinian model and our hybrid Darwinian/Lamarckian model can effectively adapt to the most unstable environment. The performance graphs of Figure 10-6 indicate that the hybrid model is clearly the most effective, followed by the Darwinian model. Note, however, that these two models utilized different reinforcement feedback values in order to maximize their initial performance in the stable environment, the hybrid's feedback being higher to benefit an initially Lamarckian nature with the Darwinian model utilizing lower values more conducive to the evolution process's response to stable conditions. It appears, then, that the hybrid's higher reinforcement feedback values are also responsible for a more effective Darwinian performance in the most unstable environment, with optimization being provided to the Darwinian model in order to adjust its reinforcement feedback towards more appropriate, higher values. All things equal, both these models should produce an equivalent performance in this particular environment.

In comparison, the remaining three models demonstrate no effectiveness at all. In fact, the Lamarckian model, so effective in the stable environments, produces the worst performance of all in response to these conditions of high instability. Clearly the inheritance of learnt knowledge, although highly beneficial in stable environments, can be detrimental to the same extent in response to highly unstable environments.

Our hybrid Darwinian/Lamarckian model, though, produces agents of a Darwinian nature in order to effectively adapt to the most unstable environments. Agents are produced utilizing genotype knowledge, created via an evolution process of selection, reproduction and mutation, and phenotype knowledge, modifiable via lifetime reinforcement learning. As a result, the agents can contain general, innate knowledge regarding the populations' experience of the environment and the capability to refine this knowledge according to the particular characteristics of their individual experience of environment.

Thus, our hybrid model, utilizing both its Darwinian and Lamarckian adaptive characteristics, demonstrates the most effective overall ability to adapt to the entire range of environments, from stable to unstable.

10.4 Conclusion

Our hybrid Darwinian/Lamarckian model clearly produced the most effective performance in the range of examined environments, from stable to unstable. By combining learning and evolution in a framework utilizing the adaptive advantages of both Lamarckian and Darwinian evolution, the model is shown to be capable of a superior adaptability. Evolution of a Lamarckian nature is utilized to produce agents that are highly adaptable towards the more stable environments, with evolution of a Darwinian nature being utilized to produce agents ideally suited towards the most unstable environments. Thus, in comparison to the performances of the other models, our hybrid model demonstrates the ability to combine their adaptive advantages to produce a superior performance, maximizing adaptability towards environments of a stable to unstable nature.

Chapter 11: Conclusion

Throughout this thesis we have presented the design, evaluation and comparison of several different learning models: evolution, reinforcement learning, Darwinian evolution and reinforcement learning, Lamarckian evolution and reinforcement learning and hybrid Darwinian/Lamarckian evolution and reinforcement learning models.

11.1 Evolution Model

Our evolution model proves to be an acceptable means of developing our agent's neural networks in the stable environment. Given limited feedback effective behaviour is evolved. That is, our agents learn to find food and avoid poison.

However, as the environment becomes more unstable the effectiveness of the model degrades to ultimately be considered ineffective in the most unstable environments.

The model is limited in its ability to adapt to such environments, since agents are evolved containing fixed knowledge via a generation or population level learning process. This results in a limited ability to unlearn incorrect behaviour and relearn correct behaviour under conditions of change.

In addition, the model demonstrates no ability to assimilate the changes and improve learning times over the course of the simulation.

Thus, our evolution model, although quite effective under stable circumstances, demonstrates no ability to evolve solutions in response to conditions of a highly unstable nature.

11.2 Reinforcement Learning Model

Our reinforcement learning model demonstrates an effective ability, superior to that of the evolution model, to produce successful agents in response to an environment demonstrating stable circumstances. Although the feedback is of a similar, limited nature as that given to the evolution model, here learning takes place during an individual's lifetime as opposed to the generation timescale of our evolution model. In addition, the feedback is utilized to target particular aspects of the agent's neural network. In comparison, our evolution model utilized fitness scores to judge agents as a whole in order to produce a new generation. As a result, the reinforcement learning model is capable of producing results of a greater effectiveness than the evolution model over the same number of generations.

However, its performance in response to increasingly unstable environments ultimately degrades to become as ineffective as that of the evolution model. Although learning in this case is applied with a greater frequency, it cannot help the model deal any more effectively with the dynamics of the most unstable environments. The reinforcement and evolution model both face the problem of learning a relationship between an object and its fitness or reinforcement value that needs to be unlearnt in a

later generation, in the extreme case, after every generation. Although the reinforcement learning model does demonstrate superior unlearning and relearning times, it demonstrates no ability to assimilate the changes and improve learning times as the simulation progresses. Thus, like the evolution model it is limited in its ability to learn and apply opposing solutions required for an effective adaptation to the unstable environments.

11.3 Darwinian Evolution and Reinforcement Learning Model

Our hybrid Darwinian evolution and reinforcement learning model combines the two previous models to produce a mechanism capable of learning at both a population and an individual level. Thus, learning occurs at a genotype level (via evolution) and phenotype level (via reinforcement learning). This allows regularities at two different timescales to be extracted, the genotype encoding information influenced by the experience of the population over many generations and the phenotype encoding information influenced by the experience of an individual during its lifetime. As a result, knowledge relating to the general characteristics of the environment over many generations can be maintained in the population's genotypes with phenotype (reinforcement) learning being utilized to adapt a particular agent to the particular characteristics of its environment.

It is shown that the Darwinian evolution and reinforcement learning model makes use of the separation of learning at a genotype and phenotype level to produce agents that can effectively deal with the most unstable environments. An analysis of the genotype reveals it to contain general information that guides the agent's learning ability. It is the innate information contained in the agent's genotype and produced by an evolution process that allows the agents to effectively utilize their individual reinforcement learning ability to discriminate between food and poison objects.

The nature of this knowledge is to produce agents that initially attempt to locate both objects, after which they are appropriately reinforced, thereby effectively learning the relationship between an object and its fitness consequences. This relationship is then "forgotten" in the following generation, with offspring agents returning to their innate knowledge. The advantage being that knowledge, now possibly incorrect, from a previous environment does not need to be unlearnt. Rather, innate knowledge that has been formed from the population's long-term experience of the environment is utilized to provide offspring agents with innate, general knowledge that can be further refined through reinforcement learning to meet the requirements of the particular situation.

The disadvantage of this model is that it is shown to be less effective in the stable environments than the reinforcement learning model and only equivalent to the evolution model if fixed reinforcements are used, while being a bit worse if the reinforcements are open to optimization. Attempting to optimize reinforcement values is found to be necessary for an effective performance if the environment decays from stable to unstable, since different degrees of learning are required in environments of a different stability. This is found to be problematic in a stable environment as it extends the solution space to a degree that is difficult for the search process to

effectively handle. The solution space terrain becomes more difficult to navigate, hampering the search process's ability to find effective solutions.

However, with a view towards adapting in an environment of changing stability it is shown that optimizing the feedback values is useful and necessary, as the degree of reinforcement required varies according to the stability of the environment. In unstable environments the evolution model produces knowledge of a general nature that requires large reinforcements to transform that knowledge into the appropriate solution. In stable environments the evolution process produces innate agent knowledge that needs only small reinforcements to result in improvements, since the evolution process is in itself capable of adapting to such environments with learning playing a smaller role.

Allowing the system to optimize reinforcements provides it with the ability to adapt the degree of individual learning required according to the demands of the environment. As a result, our evolution process can complement reinforcement learning by providing both innate, general knowledge regarding the population's experience of the environment and appropriately adjust the degree of reinforcement learning required, thereby modifying the agents' evolved knowledge according to the particular requirements of their environment.

Thus, optimizing reinforcement values in conjunction with a separation of learning at a genotype and phenotype level provides a model that can effectively produce solutions across the range of environments from stable to unstable.

11.4 Lamarckian Evolution and Reinforcement Learning Model

Our Lamarckian evolution and reinforcement learning model is shown to demonstrate adaptive characteristics that are highly effective in response to the stable environments. Selection and reproduction combined with reinforcement learning creates a model that has the ability to utilize useful knowledge produced by reinforcements, as opposed to random mutations, to accelerate the search process. Since mutations are undirected, a superior performance in stable environments can be produced using reinforcement learning as a more informative genetic operator. This also results in the Lamarckian evolution and reinforcement learning model demonstrating an exceptional ability to unlearn incorrect knowledge and relearn correct solutions in response to changes in an environment's characteristics. However, unlike the Darwinian model, it is incapable of assimilating knowledge regarding such changes in order to improve its effectiveness over the course of generations. As a result, its high levels of effectiveness ultimately and rapidly decay in the environments utilizing the highest levels of instability.

As is the case with the evolution model and the reinforcement learning model, our Lamarckian evolution and reinforcement learning model is fundamentally flawed in dealing with highly unstable environments. Solutions are produced that need to be unlearnt as soon as the environment changes. Thus, our Lamarckian evolution and reinforcement learning model is unable to adapt to the most unstable environments tested while being highly effective in dealing with the more stable environments.

Since our previous conclusions indicate that Darwinian and Lamarckian evolution models have adaptive characteristics that are ideally suited towards environments of a particular nature, work was performed to investigate the potential benefits of combining both forms of evolution.

11.5 Hybrid Darwinian/Lamarckian Evolution and Reinforcement Learning Model

Our hybrid Darwinian/Lamarckian models demonstrate a superior adaptability, appropriately adjusting the required degree of inheritance according to the conditions of the environment.

The heterogeneous population model is shown to be highly effective in adjusting its population's agents to utilize the appropriate form of inheritance, from Lamarckian in stable conditions to Darwinian in unstable conditions. Our results demonstrate that the model is capable of optimizing the percentage of either type of agent according to the characteristics of the environment and the stage of the simulation. It is shown to effectively combine the advantages of Darwinian and Lamarckian evolution models to produce a more effective hybrid capable of adapting to the entire range of our examined environmental conditions, from stable to unstable.

The Darwinian/Lamarckian connection model is shown to be the least effective of our hybrid models. Although demonstrating an adequate level of adaptability in response to the range of examined environmental conditions it cannot match the performance levels achieved by the pure Darwinian and pure Lamarckian models in their favoured conditions. It appears that the search's larger solution space, due to the model's requirement that each connection be appropriately configured (Lamarckian or Darwinian), and the brittle nature of this model hinder its ability to find effective solutions to the examined problem.

The global degree of neural network inheritance model provides the most successful performance, exceeding that of the heterogeneous population model in the most unstable conditions. Utilizing agents capable of various degrees of inheritance allows the model to more effectively move its search towards effective solutions. As a result, the appropriate degree of inheritance can be adjusted and utilized according to the stage of adaptation and the environment's level of stability.

The local degree of connection inheritance model is also shown to be effective in adapting to the environment. Although each connection's degree of inheritance must be appropriately configured, thereby extending the solution space, effective agents are produced in the entire range of environmental conditions.

Thus, all our hybrid Darwinian/Lamarckian models demonstrate an effective adaptive ability in response to the examined environmental conditions, with the heterogeneous population model and the global degree of neural network inheritance model providing the most successful results.

By combining the advantageous of both Darwinian and Lamarckian evolution we have, thus, provided several models capable of adapting to a range of environment

conditions, from stable to unstable, appropriately adjusting the required degree of inheritance in response to the nature of the environment.

In conclusion then, although our experiments were applied to a particular problem context, our results did reveal certain fundamental truths regarding evolution and reinforcement learning models.

Evolution models produce individuals utilizing fixed knowledge produced over many generations as a result of selection and variation methods. Thus, they are limited in their ability to adapt to rapidly changing conditions.

Reinforcement learning is capable of extracting regularities at a smaller timescale, since it is applied throughout an individual's lifetime. However, such learning also demonstrates a limited response to highly unstable conditions. It follows the conditions in a "short-sighted" manner, continuously learning and unlearning solutions in response to changes. Such learning requires an additional mechanism to extract long-term knowledge that transcends the "short-sightedness" of a conventional reinforcement learning model.

Darwinian evolution and reinforcement learning provides such a mechanism. Long-term, general knowledge regarding the population's experience of the environment can be extracted utilizing an evolution process. Individuals capable of reinforcement learning can refine this knowledge to meet the particular requirements of the problems they encounter. Thus, evolution in a Darwinian framework can evolve general knowledge regarding the population's experience of the environment over many generations to effectively complement a learning mechanism, thereby transcending the "short-sightedness" of such learning.

In stable conditions, where learnt knowledge remains appropriate, learning and evolution can be more successfully combined in the form of Lamarckian evolution. Here, the selection and reproduction of learnt knowledge accelerates the adaptive process. However, in response to highly unstable conditions, these adaptive characteristics result in Lamarckian evolution being given to overly emphasizing solutions regarding short-lived environmental or problem specifics. Since these specifics experience rapid change in highly unstable conditions, such solutions must be continuously unlearnt, thereby limiting a Lamarckian model's adaptive ability in response to such conditions. Darwinian evolution, on the other hand, is capable of maintaining general knowledge regarding the environment that can be appropriately refined, via learning, to meet the requirements of any particular situation.

Lastly, results achieved by our hybrid models demonstrate that these fundamental adaptive qualities can be combined to produce a superior adaptability to a range of conditions. Since these conclusions relate to the models' fundamental adaptive qualities, we are confident that they will remain appropriate regardless of any particular problem's specifics.

11.6 Contributions

Our research has provided, evaluated and compared several evolution and reinforcement learning models, in isolation and combined, in Darwinian, Lamarckian and hybrid Darwinian/Lamarckian frameworks. As a result, it has contributed to an understanding of the interaction between evolution and learning, with a particular emphasis on the adaptive qualities of Darwinian and Lamarckian evolution. In addition, we have demonstrated how such adaptive qualities can be combined to provide more effective hybrid models, utilizing the adaptive advantages of both Darwinian and Lamarckian inheritance in response to the situation's requirements.

In terms of related work, our research has validated previous results indicating the nature of Lamarckism to favour stable environments and the nature of Darwinism to favour unstable environments [Sasaki and Tokoro, 1997] [Sasaki and Tokoro, 1999].

In addition, we have provided and evaluated the use of inheritance optimization mechanisms, resulting in the creation of hybrid Darwinian/Lamarckian models capable of effectively adjusting and utilizing the appropriate degree of inheritance according to an environments degree of stability, ranging from stable to unstable. Since different environments or problems have been shown to be suited towards either Lamarckism or Darwinism [Houck, Joines, Kay, and Wilson, 1997] [Imada and Araki, 1996] [Julstrom, 1999] [Sasaki and Tokoro, 1997] [Sasaki and Tokoro, 1999] [Whitley, Gordon and Mathias, 1994], such hybrid models can prove to be highly useful.

Our extensive investigation revealed that the hybrid Darwinian/Lamarckian models are more effective than our pure Darwinian and Lamarckian models in response to the conditions of the examined environments, optimizing the degree of Lamarckism or Darwinism according to the requirements of the environment.

Since most current evolution models utilize either Darwinian or Lamarckian evolution, our approach provides a contribution to the field of evolutionary computation by suggesting that hybrid models, utilizing the advantageous of both processes, can result in the creation of models demonstrating a superior level of adaptability. Although previous work has been performed utilizing various, specific degrees of inheritance, from 0% inheritance (100% Darwinian) to 100% inheritance (100% Lamarckian) [Houck, Joines, Kay, and Wilson, 1997] [Sasaki and Tokoro, 1999], we investigate and provide the means for an optimization mechanism that can appropriately adjust the degree of inheritance in response to the environment's conditions. As a result, a highly adaptive evolution model is created with the capability to utilize the adaptive advantages of both Lamarckian and Darwinian evolution according to the demands of the environment.

Thus, our research results indicate that, unlike conventional evolution models, an advantageous combination of both Darwinian and Lamarckian adaptive characteristics can result in the creation of more effective hybrids, utilizing the appropriate degree of inheritance according to the nature of the problem being solved.

11.7 Future Work

The increasing complexity of computer systems is going to require new adaptive mechanisms for dealing with the problems they pose. Evolutionary computation is capable of providing such adaptability, creating emergent solutions in response to the problem environment. Since complex systems are likely to be highly dynamic, methods invoking both Darwinian and Lamarckian inheritance will be required. Our work is a step in this direction.

Future work will involve the refinement and extension of such techniques. A particularly interesting outcome of this development could be an accurate modeling of the interaction between meme and gene evolution, memetic evolution being an example of a process demonstrating Lamarckian characteristics. This could lead to new adaptive mechanisms utilizing the advantages of both memetic and genetic evolution to solve the problems posed by highly complex environments.

References

[Ackley and Littman, 1990] Ackley, D.H. and Littman, M.L., "Generalization and Scaling in Reinforcement Learning", In Touretsky, D.S., (Ed.), *Advances in Neural Information Processing Systems 2*, pp. 550-557, San Matco, CA, Morgan Kaufmann, 1990.

[Balakrishnan and Honavar, 1995a] Balakrishnan, K., Honavar, V., "Evolutionary Design of Neural Architectures", Technical Report CS TR 95-01, Department of Computer Science, Iowa State University, Ames, IA, 1995a.

[Balakrishnan and Honavar, 1995b] Balakrishnan, K., Honavar, V., "Properties of Genetic Representations of Neural Architectures", In Proceedings of the World Congress on Neural Networks, 1995b.

[Baldwin, 1896] Baldwin, J.M.,"A new factor in evolution", *American Naturalist*, 30:441-451, 1896.

[Barto and Anandan, 1985] Barto, A.G., Anandan, P., "Pattern recognizing stochastic learning automata", *IEEE Transactions on Systems, Man and Cybernetics*, 1985.

[Barto, Sutton and Anderson, 1983] Barto, A.G., Sutton, R.S., Anderson, C.W., "Neuronlike adaptive elements that can solve difficult learning control problems", *IEEE Transactions on Systems, Man and Cybernetics*, SMC - 13(5), 834-846, 1983.

[Baxter, 1992] Baxter, J., "The evolution of learning algorithms for artificial neural networks", In Green, D. and Bossomaier, T., (editors) *Complex Systems*, pp. 313-326, IOS Press, Amsterdam, 1992.

[Belew, McInerney and Schraudolph, 1990] Belew, R.K., McInerney, J., and Schraudolph, N.N., "Evolving networks: Using the genetic algorithm with connectionist learning", CSE Technical Report #CS90-174, University of California, San Diego, 1990.

[Campbell, 1980] Campbell, D., "Evolutionary Epistemology", In *The Phylosophy of Karl Popper*, Paul Arthur Schlipp, Open Court Publ. (La Salle, Illinois), 1980.

[Cristianini, 1995] Cristianini, N., "Evolution and Learning: An Epistimological Perspective", University of Trieste, 1995.

[Fogel, 1997] Fogel, D.B., "The Advantages of Evolutionary Computation", In BCEC97, Springer, 1997.

[Fogel, Owens and Walsh, 1966] Fogel, L.J., Owens, A.J. and Walsh, M.J., "Artificial Intelligence Though Simulated Evolution", John Wiley and Sons, New York, 1966.

[Galar, 1985] Galar, R., "Handicapped Individua in Evolutionary Processes", *Biological Cybernetics*, Vol. 53, pp. 1-9, 1985.

[Grefenstette, 1986] Grefenstette, J.J., "Optimization of Control Parameters for Genetic Algorithms", *IEEE Transactions on Systems, Man and Cybernetics*, Vol. SMC-16:1, pp. 122-128, 1986.

[Heitkotter and Beasly, 1998] Heitkotter, J. and David, B. (editors), "The Hitch-Hiker's Guide to Evolutionary Computation: A list of Frequently Asked Questions (FAQ)", USENET: comp.ai.genetic, 1998.

[Hinton and Nowlan, 1987] Hinton, G.E. and Nowlan, S.J., "How learning can guide evolution", *Complex Systems*, 1:495-502, 1987.

[Holland, 1975] Holland, J.H., "Adaptation in Natural and Artificial Systems", University of Michigan Press, Ann Arbor, 1975.

[Houck, Joines, Kay, and Wilson, 1997] Houck, C., Joines, J.A., Kay, M.G. and Wilson, J.R., "Empirical investigation of the benefits of partial Lamarckianism", *Evolutionary Computation*, v.5, n.1, pp.31- 60, 1997.

[Julstrom, 1999] Julstrom, B.A., "Comparing Darwinian, Baldwinian, and Lamarckian Search In a Genetic Algorithm for the 4-Cycle Problem", In *Late Breaking Papers at the 1999 Genetic and Evolutionary Computation Conference* (GECCO'99) (Scott Brave and Annie S. Wu. Eds.), Orlando, FL, pp.134-138, July 1999.

[Imada and Araki, 1996] Imada, A. and Araki, K., "Lamarckian evolution of associative memory", In *Proceedings of 1996 IEEE The Third International Conference on Evolutionary Computation* (ICEC-96), pages 676--680, 1996.

[Klopf, 1972] Klopf, A.H., "Brain function and adaptive systems – A heterostatic theory", Technical Report AFCRL-72-0164, Air Force Cambridge Research Laboratories, Bedford, MA, 1972.

[Littman, 1995] Littman, M.L., "Simulations combining evolution and learning", In Belew, R.K. and Mitchell, M. (editors), *Adaptive Individuals in Evolving Populations: Models and algorithms*, Santa Fe Institute Studies in the Sciences of Complexity, Addison Wesley, Reading, MA, 1995.

[Littman and Ackley, 1991] Littman, M.L. and Ackley, D.H., "Adaptation in constant utility non-stationary environments", In Belew, R.K. and Booker, L.B. (editors), *Proceedings of the 4th International Conference on Genetic Algorithms*, 1991.

[Mayley, 1997] Mayley, G., "Guiding or hiding", In the *Proceedings of the Fourth European Conference on Artificial Life* (ecal97). Husbands, P. and Harvey, I. (editors.), 1997.

[Michalewicz, 1996] Michalewicz, Z., “Genetic Algorithms + Data Structures = Evolution Programs”, Springer-Verlag, New York, 3rd ed., 1996.

[Mitchell and Forrest, 1994] Mitchell, M. and Forrest, S., “Genetic algorithms and artificial life”, *Artificial Life*, 1(3), 1994.

[Nolfi and Parisi, 1997] Nolfi S. and Parisi D., “Neural networks in an artificial life perspective”, In Gerstner, W., Germond, A., Hasler, M. and Nicoud, J.D., (editors), *Artificial Neural Networks* (ICANN97). *Proceedings of the 7th International Conference on Artificial Neural Networks*. Berlin: Springer Verlag, pp. 733-738, 1997.

[Popper, 1985] Popper K., “Thought and Experience and Evolutionary Epistemology”, in *Atti del convegno Che cos'è il pensiero? Unità dell'essere*, Accademia Nazionale dei Lincei, 1985.

[Rumelhart, Hinton and Williams, 1986] Rumelhart, D.E., Hinton, G.E., & Williams, R.J., “Learning representations by backpropagating errors”, *Nature*, 323, 533—536, 1986.

[Sasaki and Tokoro, 1997] Sasaki, T. and Tokoro, M., “Adaptation toward changing environments: Why Darwinian in nature?”, In *4th European Conference on Articial Life* (ECAL97), pages 145-153, 1997.

[Sasaki and Tokoro, 1999] Sasaki, T. and Tokoro, M., “Evolving Learnable Neural Networks under Changing Environments with Various Rates of Inheritance of Acquired Characters: Comparison between Darwinian and Lamarckian Evolution”, *Artificial Life* 5(3): 203-223, 1999.

[Schmidhuber, 1989] Schmidhuber, J., “A local learning algorithm for dynamic feedforward and recurrent networks”, *Connection Science*, 1:403-412, 1989.

[Schwefel, 1981] Schwefel, H.-P., “Numerical Optimization for Computer Models”, John Wiley, Chichester, UK, 1981.

[Sims, 1991] Sims, K., “Artificial Evolution for Computer Graphics”, *Computer Graphics* 25(4), 319-328, 1991.

[Singh and Sutton, 1996] Singh, S.P. and Sutton, R.S., “Reinforcement learning with replacing eligibility traces”, *Machine Learning*, 22(1), 1996.

[Sutton, 1984] Sutton, R.S., “Temporal Credit Assignment in Reinforcement Learning”, PhD thesis, University of Massachusetts, Amherst, MA, 1984.

[Sutton, 1988] Sutton, R.S., “Learning to predict by the method of temporal difference”, *Machine Learning*, 3(1), 9-44, 1988.

[Todd and Miller, 1991] Todd, P.M. and Miller, G.F., “Exploring adaptive agency II: Simulating the evolution of associative learning”, In J.-A. Meyer and S. W. Wilson (editors), *From animals to animats: Proceedings of the First International Conference on Simulation of Adaptive Behaviour*, pages 306--315, Cambridge, MA, MIT Press/Bradford Books, 1991.

[Whitley, Gordon and Mathias, 1994] Whitley, D., Gordon, S. and Mathias, K., “Larmarckian evolution, the Baldwin effect and function optimization”, In Y. Davidor, H., Schwefel, H.-P. and R. Manner (editors), *Parallel Problem Solving from Nature*-PPSN III, pp. 6--15. Springer-Verlag, 1994.

Design, Evaluation and Comparison of Evolution and Reinforcement Learning Models.

Appendix

Clinton Brett Mclean

April 2001

Contents

Appendix A

Barometer Selection With Replacement for Mutation

Maximum Average Fitness Scores

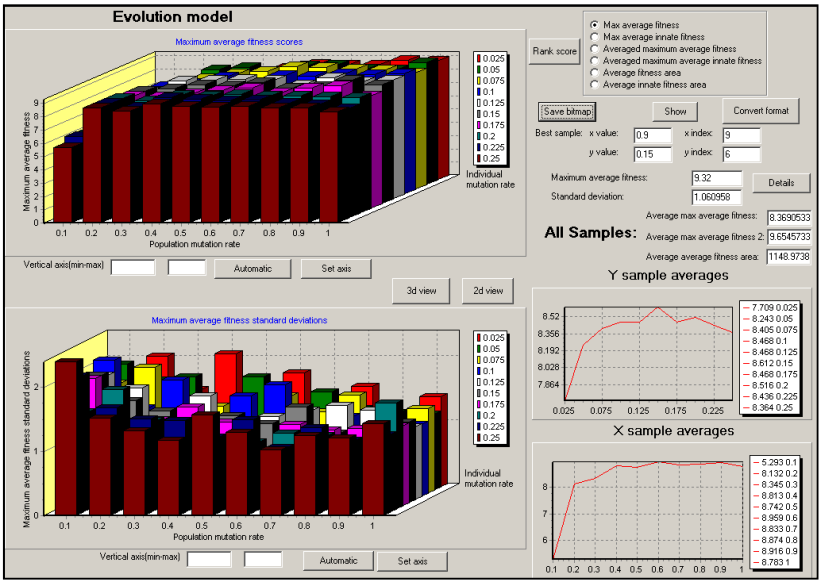


Figure A-1 3d perspective of maximum average fitness scores produced using population mutation rates in the range 0.1 to 1 and individual mutation rates in the range 0.025 to 0.25.

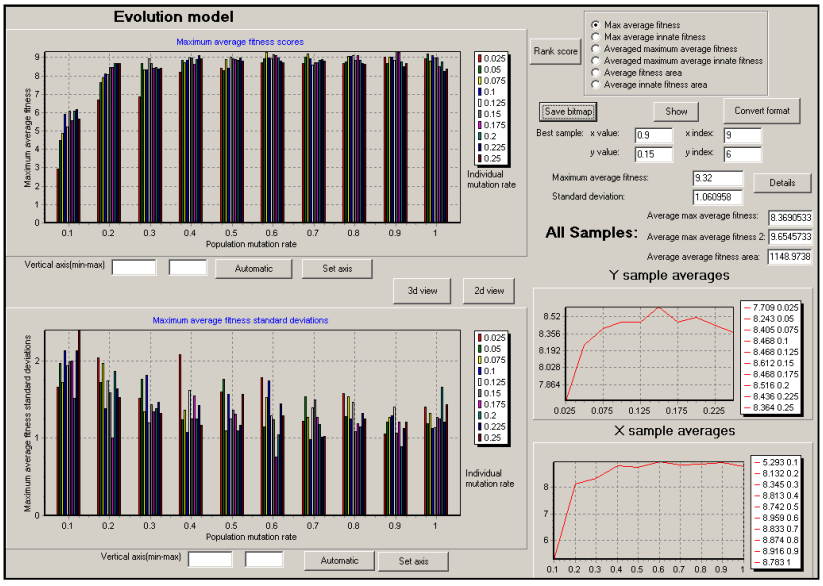


Figure A-2 2d perspective of maximum average fitness scores produced using population mutation rates in the range 0.1 to 1 and individual mutation rates in the range 0.025 to 0.25.

Averaged Maximum Average Fitness Scores

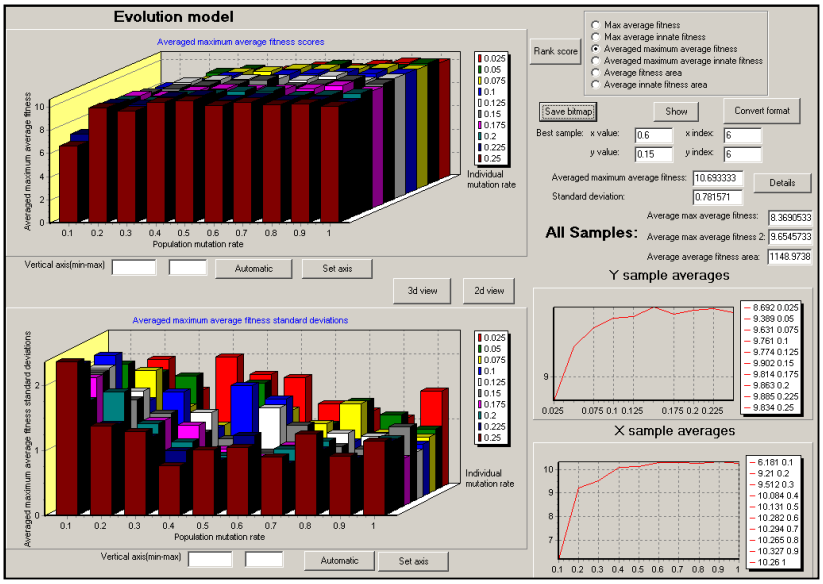


Figure A-3 3d perspective of averaged maximum average fitness scores produced using population mutation rates in the range 0.1 to 1 and individual mutation rates in the range 0.025 to 0.25.

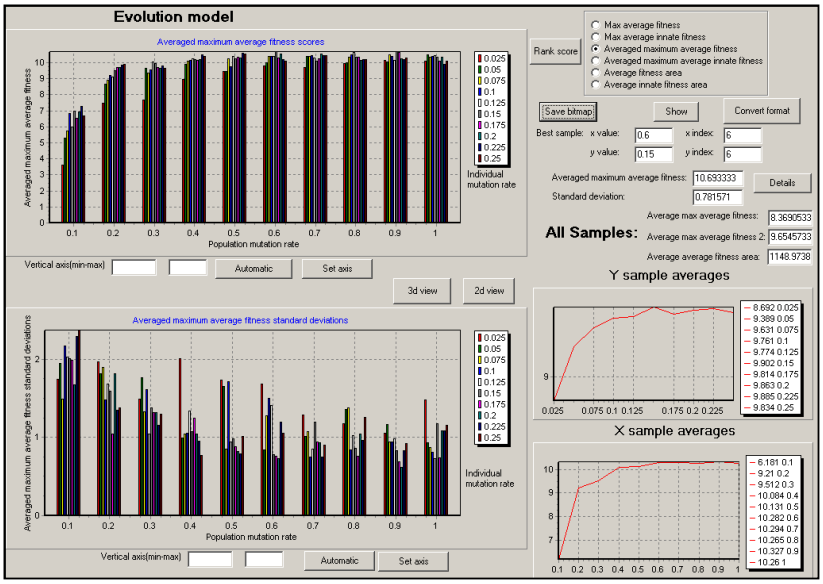


Figure A-4 2d perspective of averaged maximum average fitness scores produced using population mutation rates in the range 0.1 to 1 and individual mutation rates in the range 0.025 to 0.25.

Average Fitness Area Scores

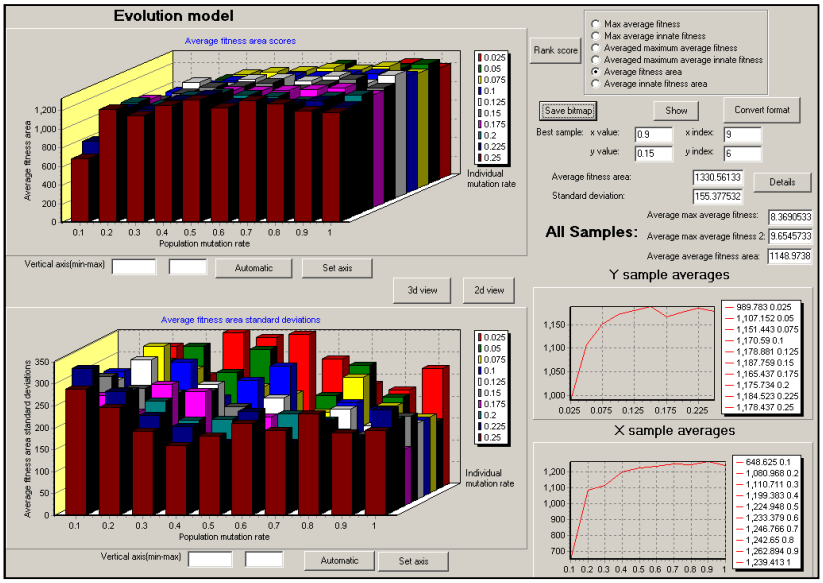


Figure A-5 3d perspective of average fitness area scores produced using population mutation rates in the range 0.1 to 1 and individual mutation rates in the range 0.025 to 0.25.

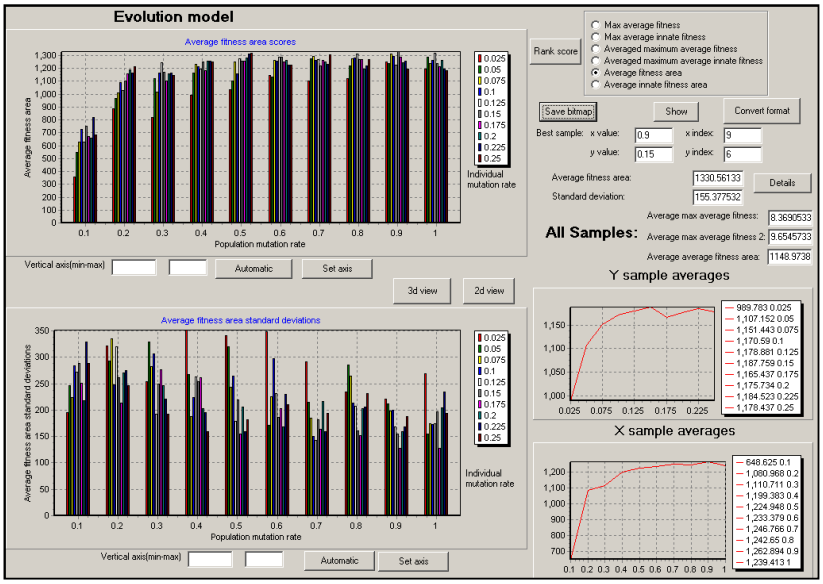


Figure A-6 2d perspective of average fitness area scores produced using population mutation rates in the range 0.1 to 1 and individual mutation rates in the range 0.025 to 0.25.

Barometer Selection Without Replacement for Mutation

Maximum Average Fitness Scores

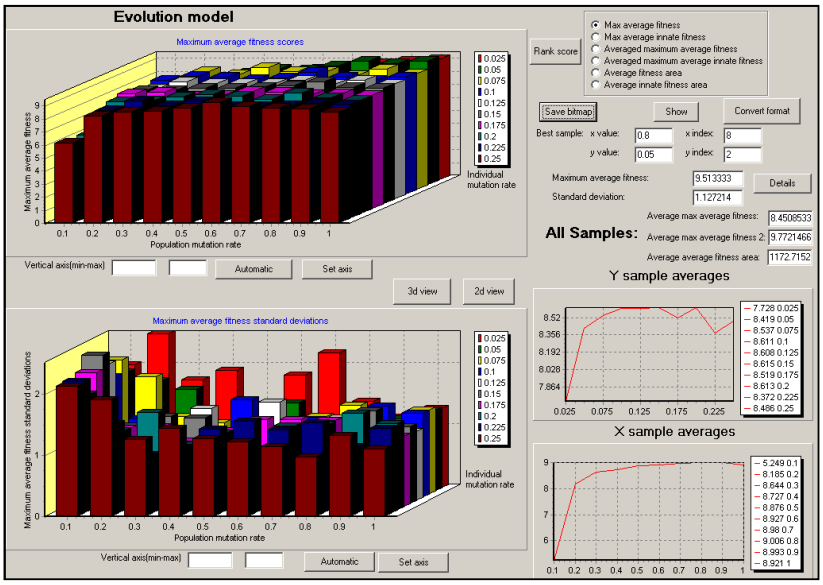


Figure A-7 3d perspective of maximum average fitness scores produced using population mutation rates in the range 0.1 to 1 and individual mutation rates in the range 0.025 to 0.25.

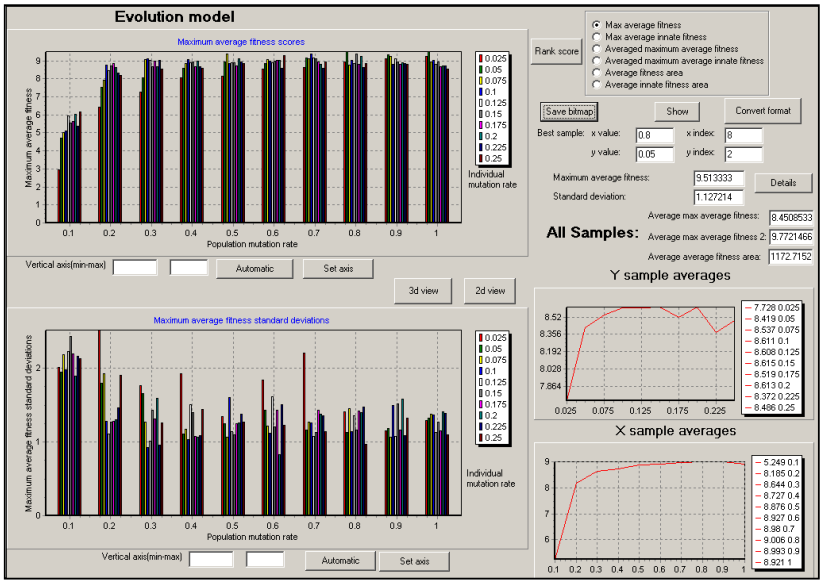


Figure A-8 2d perspective of maximum average fitness scores produced using population mutation rates in the range 0.1 to 1 and individual mutation rates in the range 0.025 to 0.25.

Averaged Maximum Average Fitness Scores

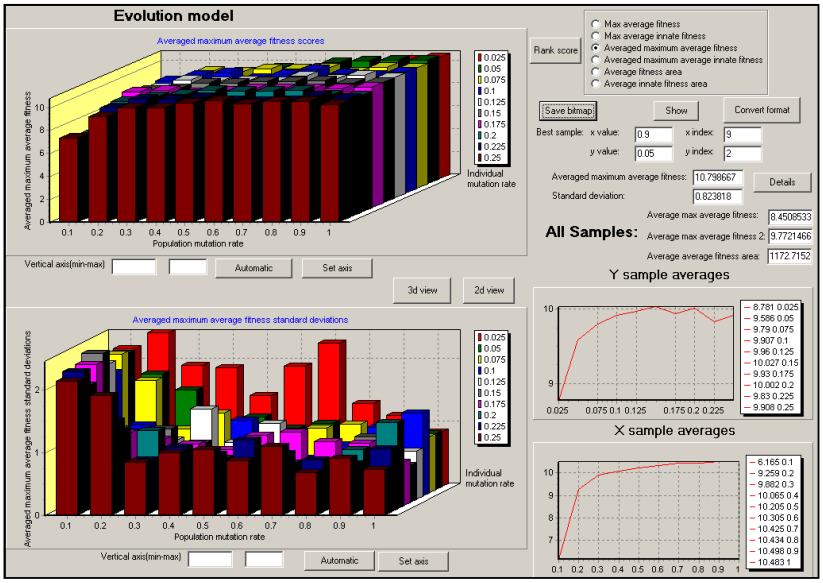


Figure A-9 3d perspective of averaged maximum average fitness scores produced using population mutation rates in the range 0.1 to 1 and individual mutation rates in the range 0.025 to 0.25.

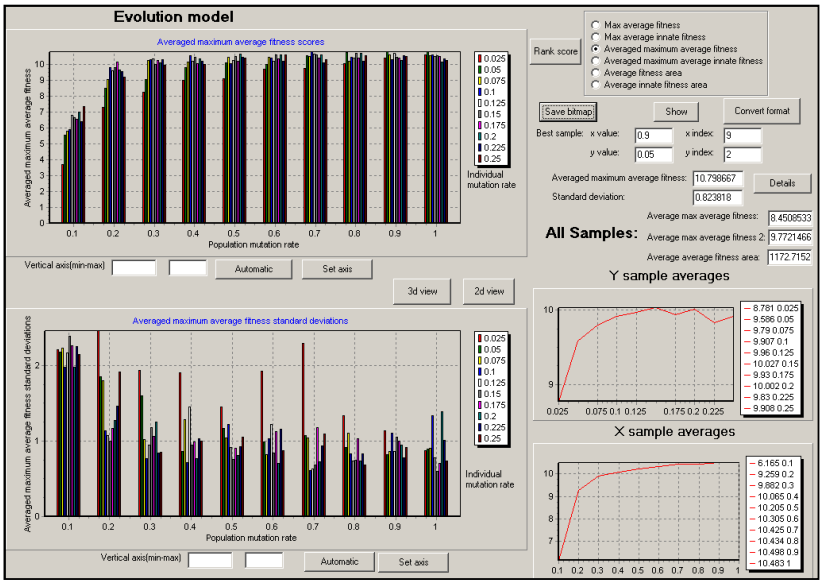


Figure A-10 2d perspective of averaged maximum average fitness scores produced using population mutation rates in the range 0.1 to 1 and individual mutation rates in the range 0.025 to 0.25.

Average Fitness Area Scores

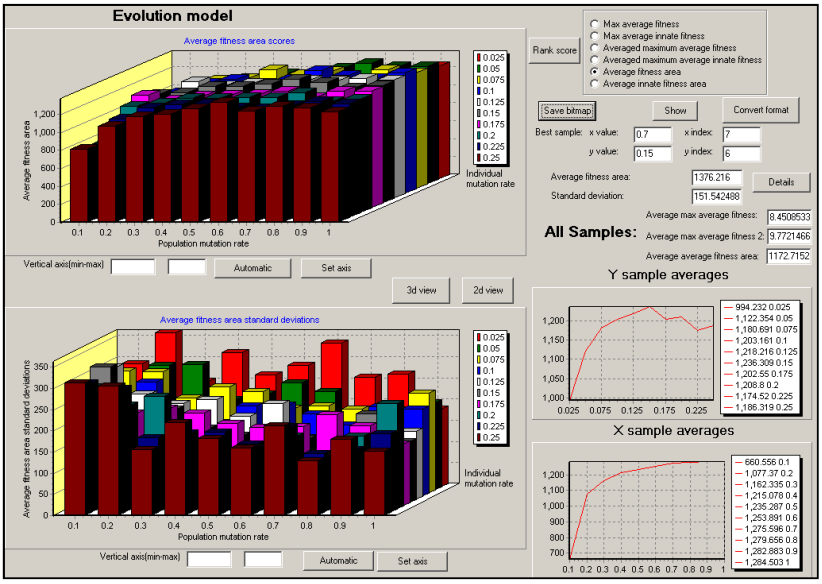


Figure A-11 3d perspective of average fitness area scores produced using population mutation rates in the range 0.1 to 1 and individual mutation rates in the range 0.025 to 0.25.

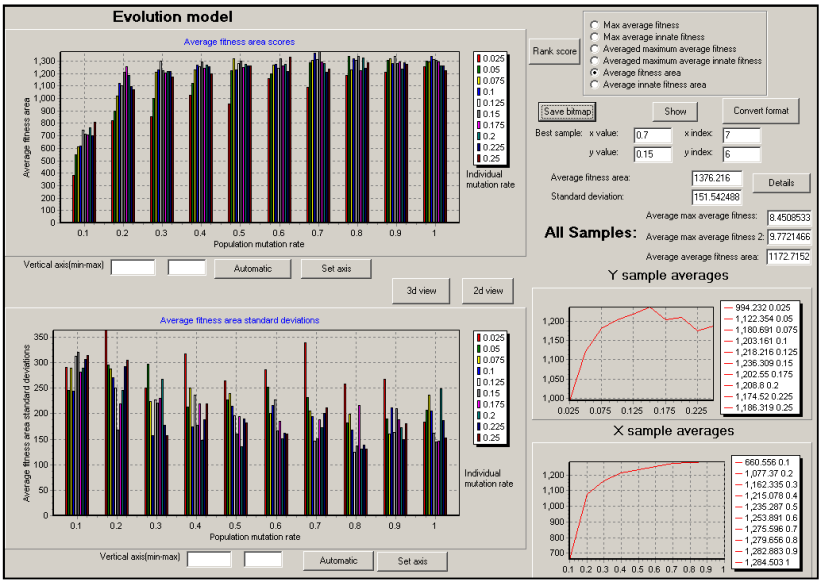


Figure A-12 2d perspective of average fitness area scores produced using population mutation rates in the range 0.1 to 1 and individual mutation rates in the range 0.025 to 0.25.

Roulette Selection With Replacement for Mutation

Maximum Average Fitness Scores

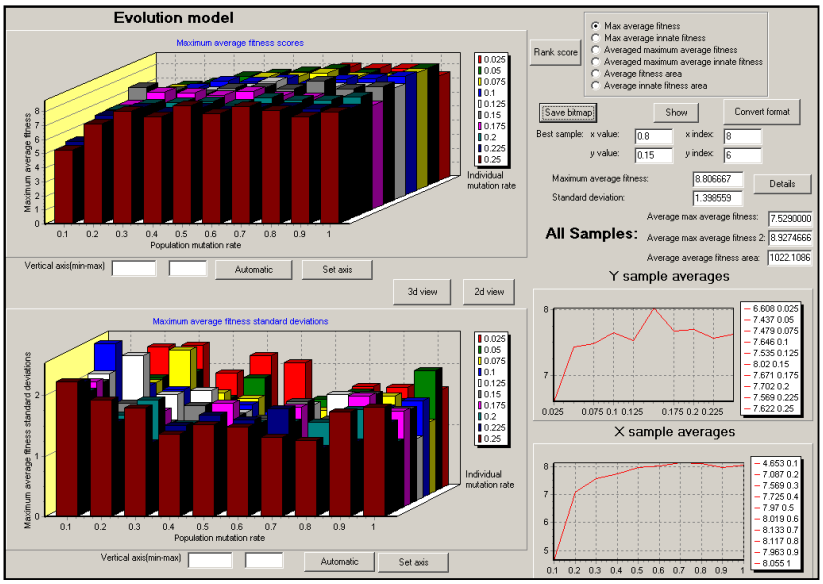


Figure A-13 3d perspective of maximum average fitness scores produced using population mutation rates in the range 0.1 to 1 and individual mutation rates in the range 0.025 to 0.25.

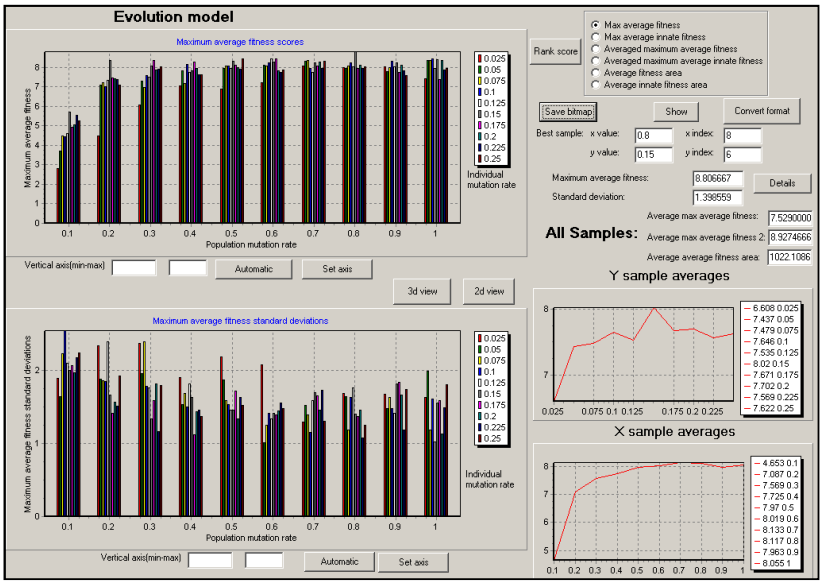


Figure A-14 2d perspective of maximum average fitness scores produced using population mutation rates in the range 0.1 to 1 and individual mutation rates in the range 0.025 to 0.25.

Averaged Maximum Average Fitness Scores

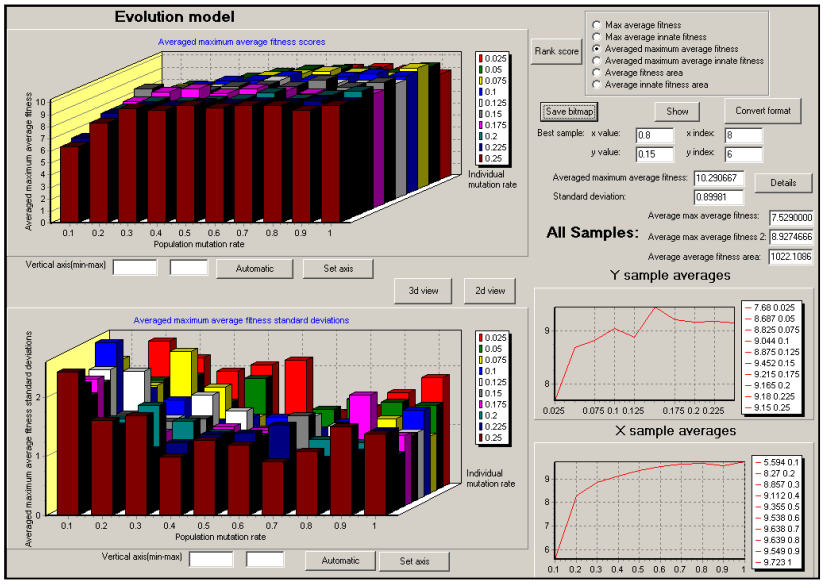


Figure A-15 3d perspective of averaged maximum average fitness scores produced using population mutation rates in the range 0.1 to 1 and individual mutation rates in the range 0.025 to 0.25.

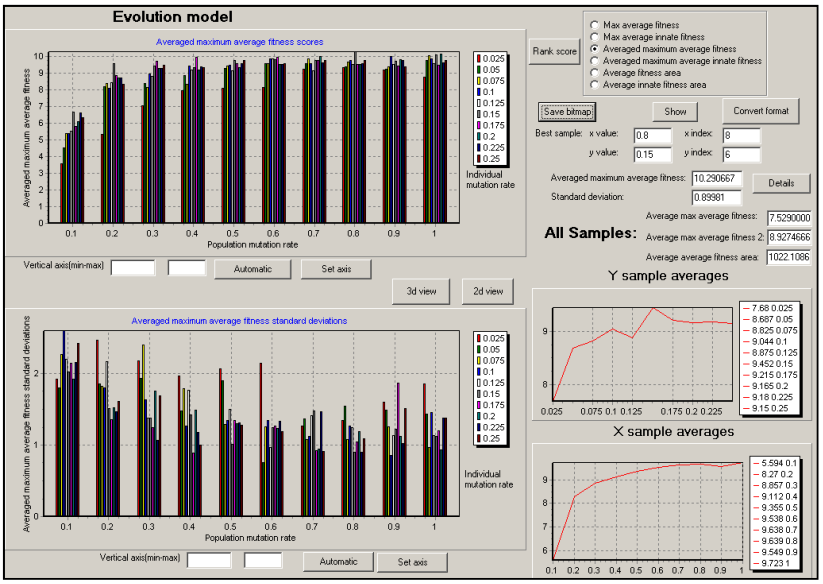


Figure A-16 2d perspective of averaged maximum average fitness scores produced using population mutation rates in the range 0.1 to 1 and individual mutation rates in the range 0.025 to 0.25.

Average Fitness Area Scores

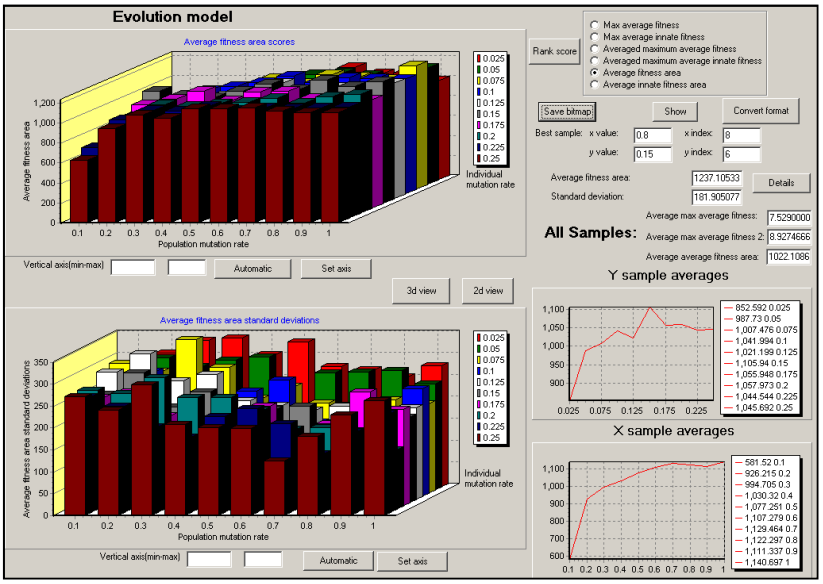


Figure A-17 3d perspective of average fitness area scores produced using population mutation rates in the range 0.1 to 1 and individual mutation rates in the range 0.025 to 0.25.

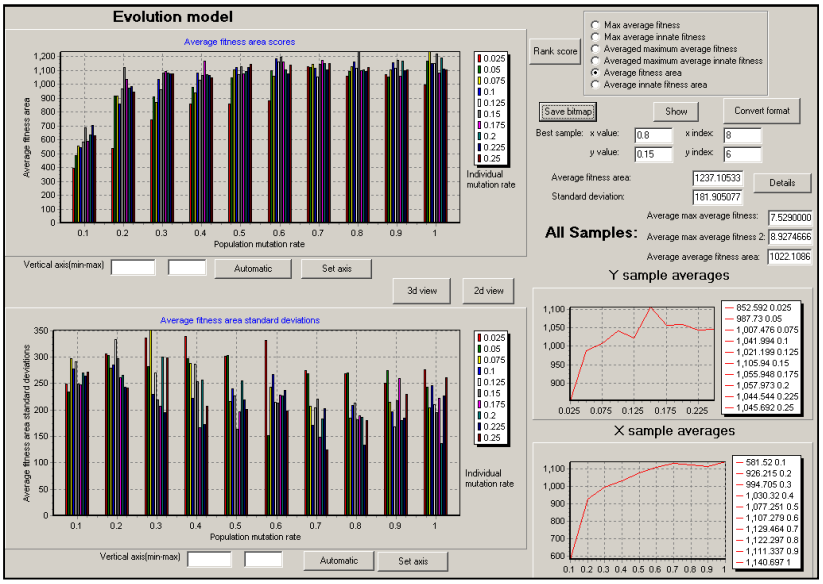


Figure A-18 2d perspective of average fitness area scores produced using population mutation rates in the range 0.1 to 1 and individual mutation rates in the range 0.025 to 0.25.

Roulette Selection Without Replacement for Mutation

Maximum Average Fitness Scores

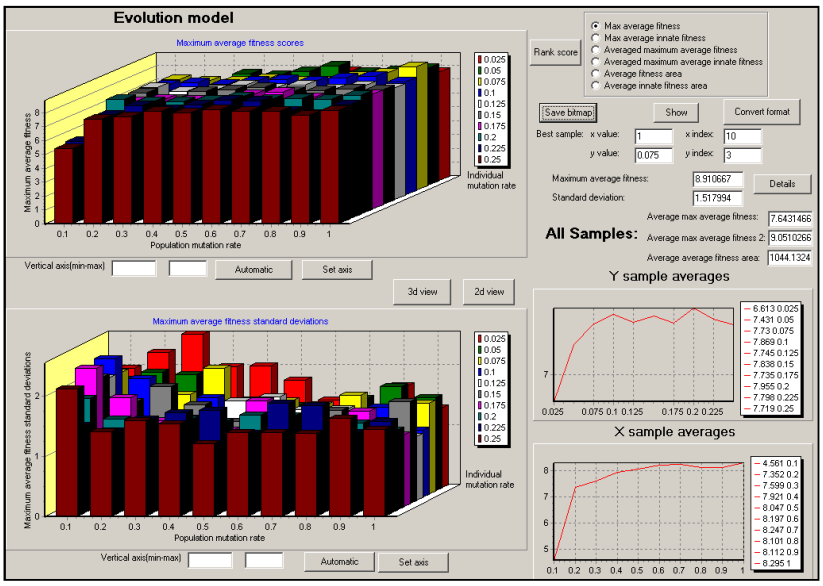


Figure A-19 3d perspective of maximum average fitness scores produced using population mutation rates in the range 0.1 to 1 and individual mutation rates in the range 0.025 to 0.25.

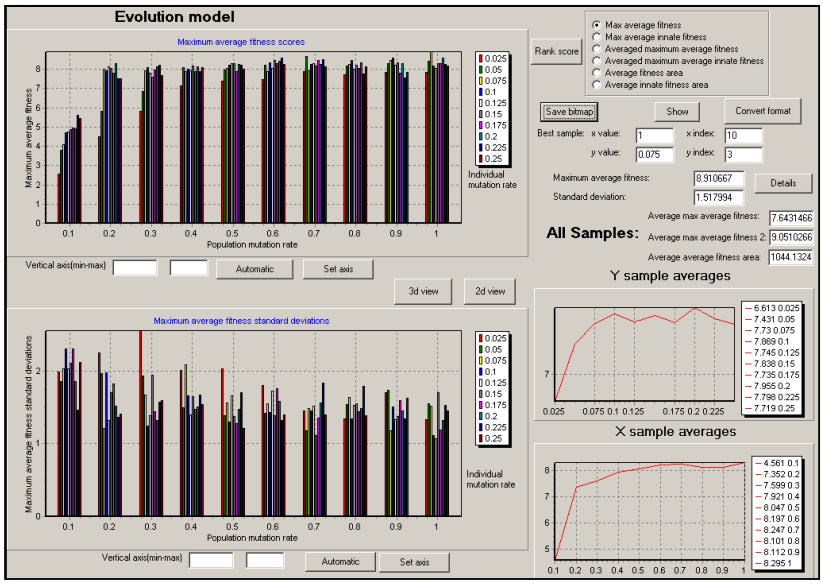


Figure A-20 2d perspective of maximum average fitness scores produced using population mutation rates in the range 0.1 to 1 and individual mutation rates in the range 0.025 to 0.25.

Averaged Maximum Average Fitness Scores

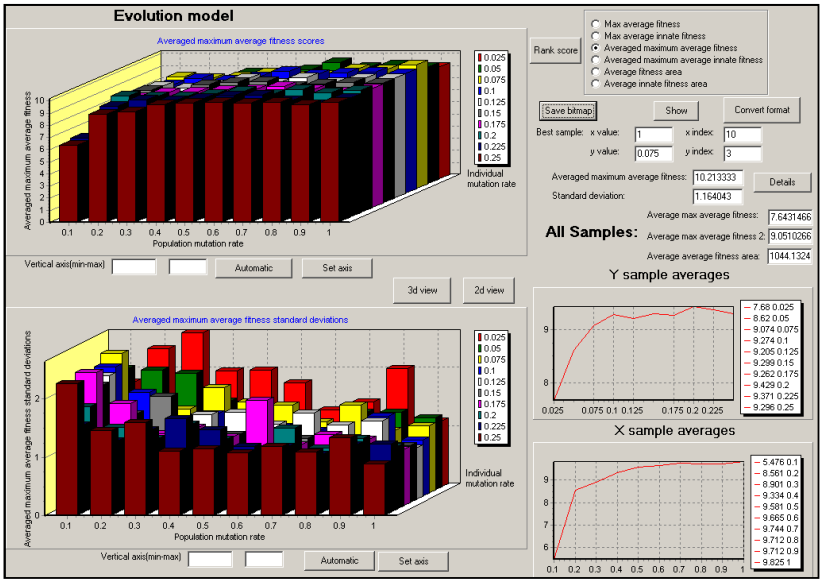


Figure A-21 3d perspective of averaged maximum average fitness scores produced using population mutation rates in the range 0.1 to 1 and individual mutation rates in the range 0.025 to 0.25.

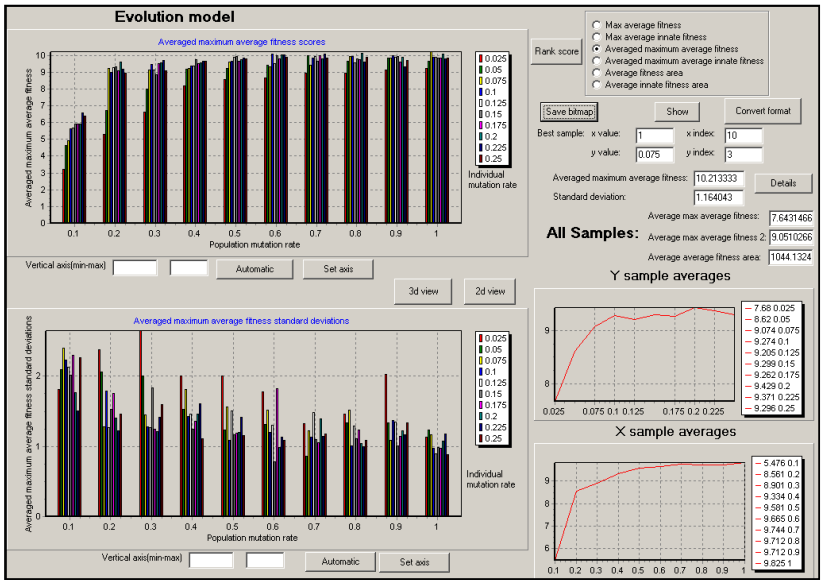


Figure A-22 2d perspective of averaged maximum average fitness scores produced using population mutation rates in the range 0.1 to 1 and individual mutation rates in the range 0.025 to 0.25.

Average Fitness Area Scores

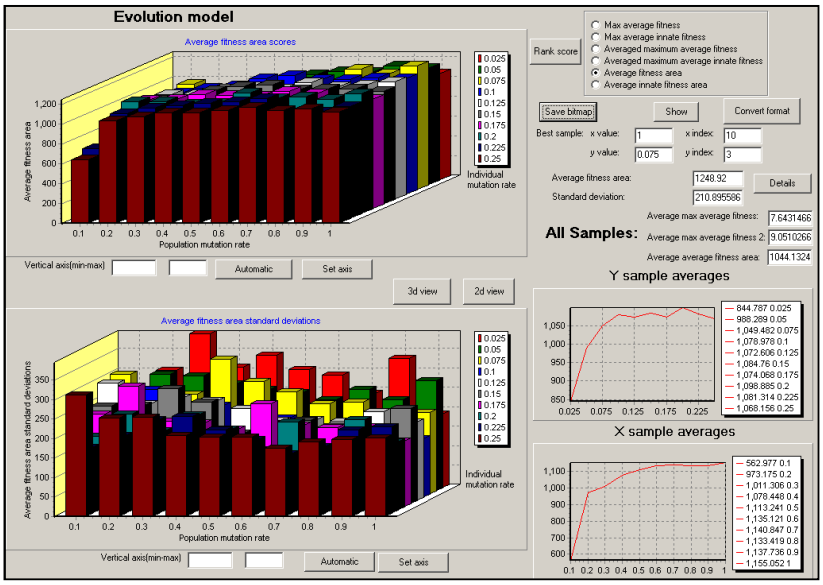


Figure A-23 3d perspective of average fitness area scores produced using population mutation rates in the range 0.1 to 1 and individual mutation rates in the range 0.025 to 0.25.

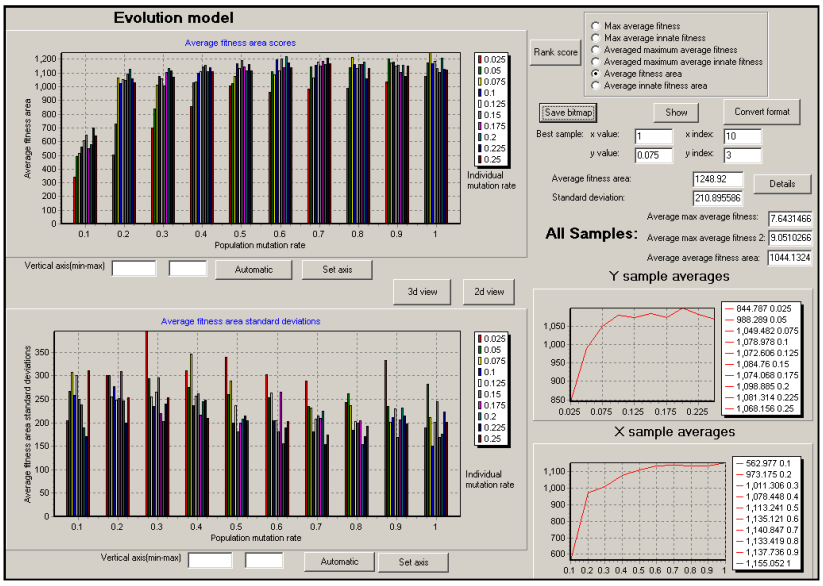


Figure A-24 2d perspective of average fitness area scores produced using population mutation rates in the range 0.1 to 1 and individual mutation rates in the range 0.025 to 0.25.

Appendix B

Barometer Selection Without Replacement for Mutation and Self-optimization Utilizing Different Degrees of Change

Maximum Average Fitness Scores

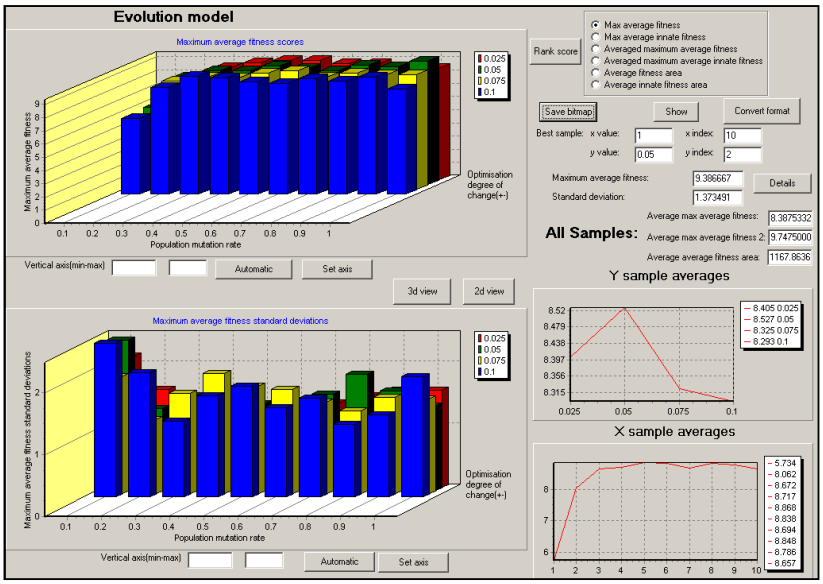


Figure B-1 3d perspective of maximum average fitness scores produced using population mutation rates in the range 0.1 to 1 allowing the individual mutation rate to self-optimize with degrees of change in the range 0.025 to 0.1.

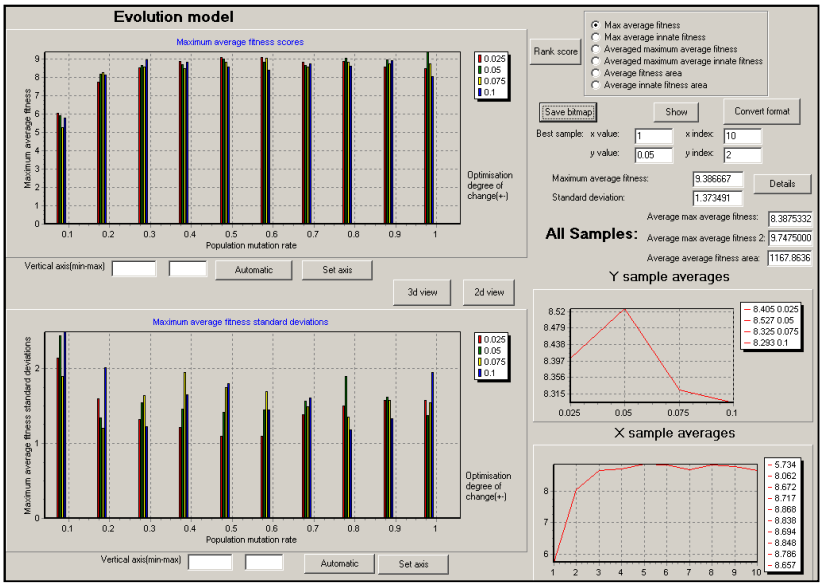


Figure B-2 2d perspective of maximum average fitness scores produced using population mutation rates in the range 0.1 to 1 allowing the individual mutation rate to self-optimize with degrees of change in the range 0.025 to 0.1.

Averaged Maximum Average Fitness Scores

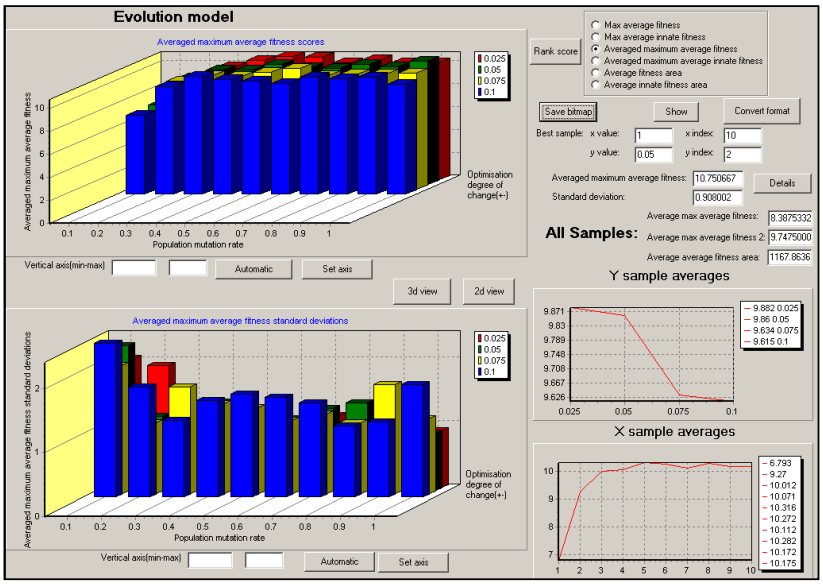


Figure B-3 3d perspective of averaged maximum average fitness scores produced using population mutation rates in the range 0.1 to 1 allowing the individual mutation rate to self-optimize with degrees of change in the range 0.025 to 0.1.

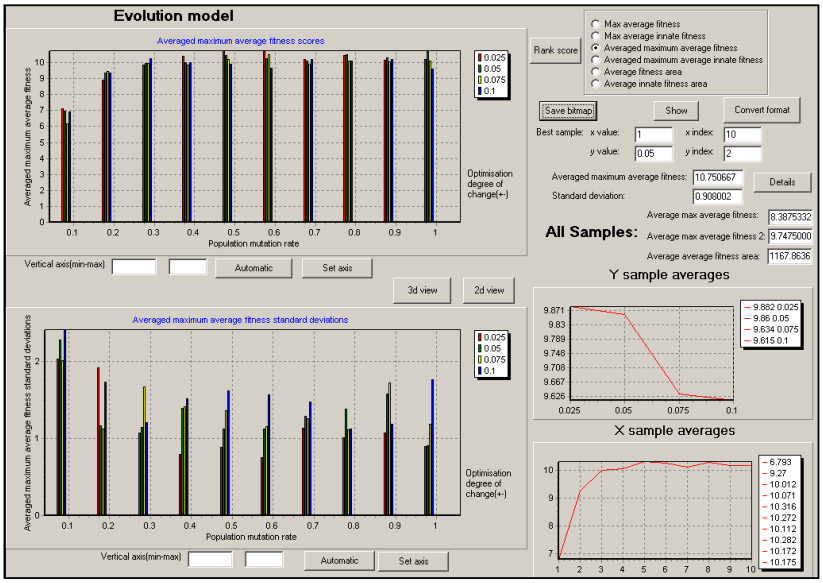


Figure B-4 2d perspective of averaged maximum average fitness scores produced using population mutation rates in the range 0.1 to 1 allowing the individual mutation rate to self-optimize with degrees of change in the range 0.025 to 0.1.

Average Fitness Area Scores

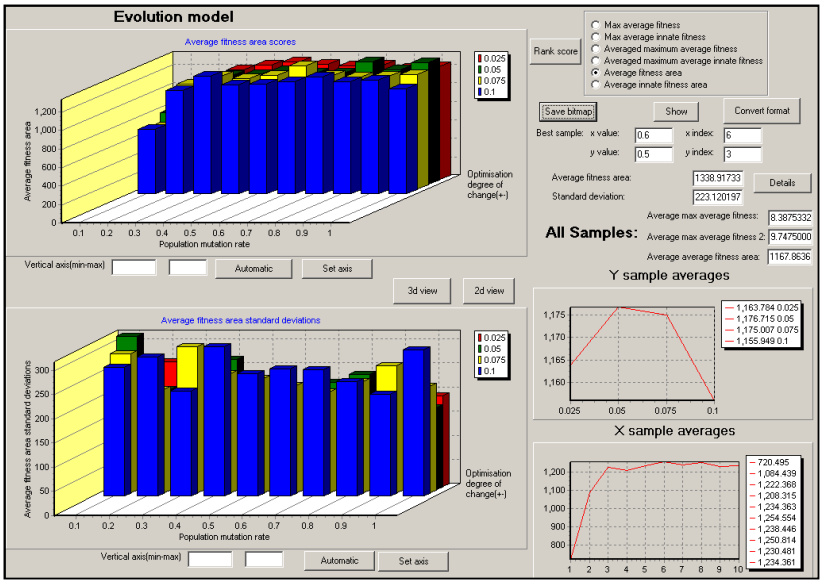


Figure B-5 3d perspective of average fitness area scores produced using population mutation rates in the range 0.1 to 1 allowing the individual mutation rate to self-optimize with degrees of change in the range 0.025 to 0.1.

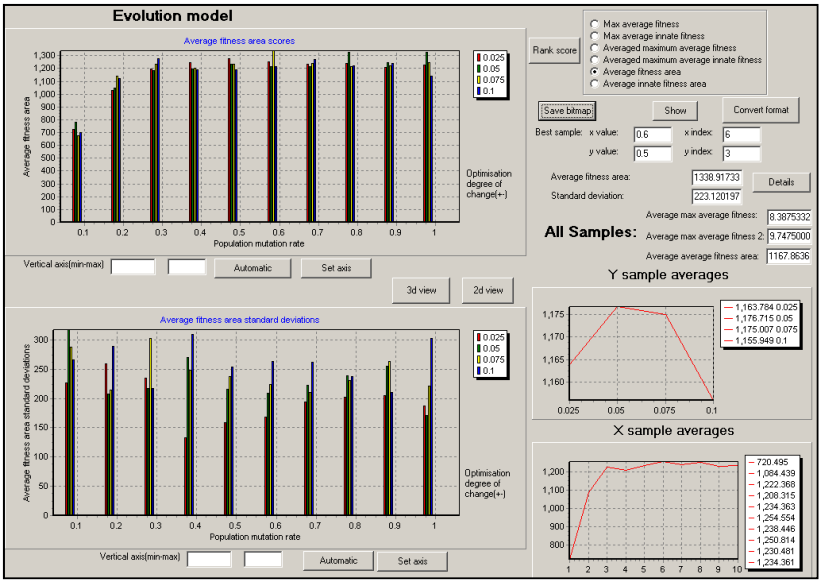


Figure B-6 2d perspective of average fitness area scores produced using population mutation rates in the range 0.1 to 1 allowing the individual mutation rate to self-optimize with degrees of change in the range 0.025 to 0.1.

Appendix C

Barometer Selection Without Replacement for Mutation and Self-optimization Utilizing Different Initial Starting Rates

Maximum Average Fitness Scores

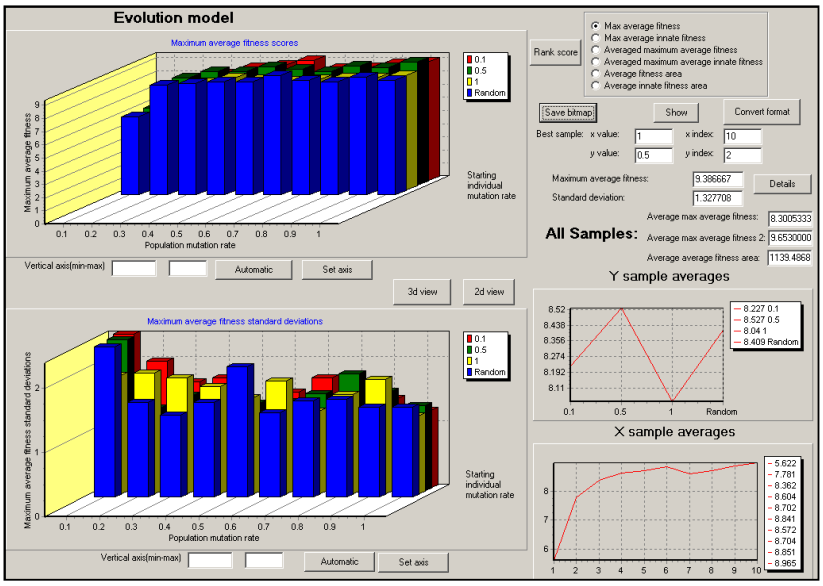


Figure C-1 3d perspective of maximum average fitness scores produced using population mutation rates in the range 0.1 to 1 allowing the individual mutation rate to self-optimize starting from rates of 0.1, 0.5, 1 and a random initialization.

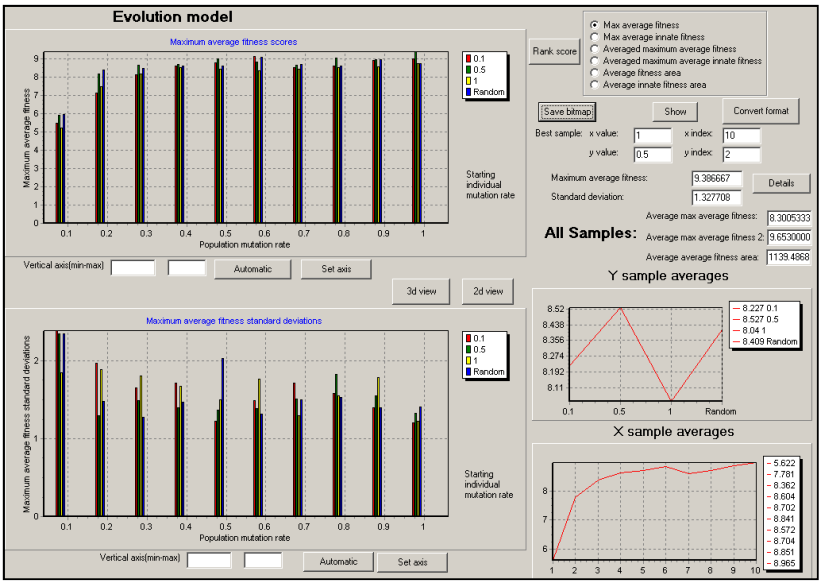


Figure C-2 2d perspective of maximum average fitness scores produced using population mutation rates in the range 0.1 to 1 allowing the individual mutation rate to self-optimize starting from rates of 0.1, 0.5, 1 and a random initialization.

Averaged Maximum Average Fitness Scores

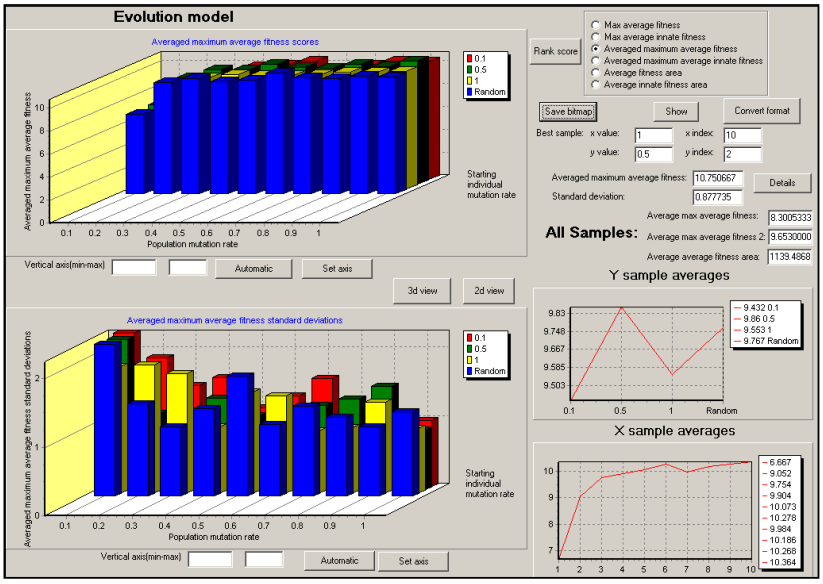


Figure C-3 3d perspective of averaged maximum average fitness scores produced using population mutation rates in the range 0.1 to 1 allowing the individual mutation rate to self-optimize starting from rates of 0.1, 0.5, 1 and a random initialization.

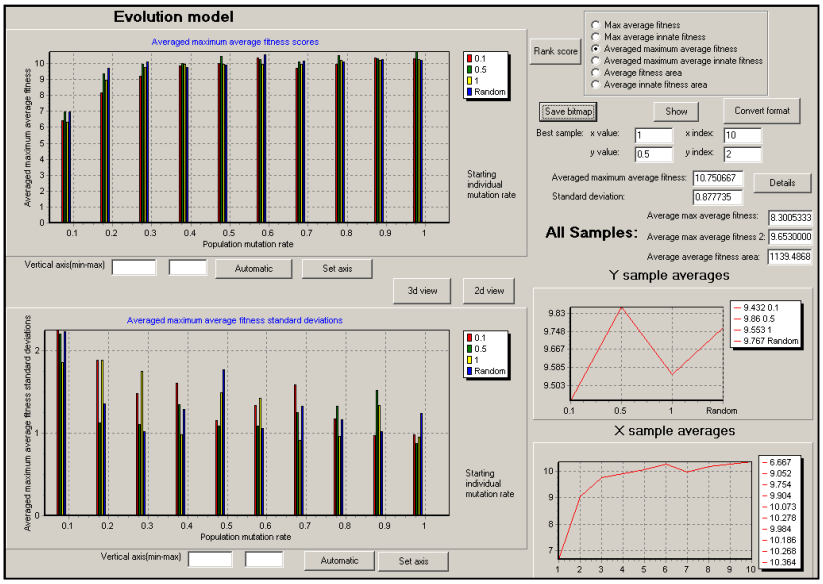


Figure C-4 2d perspective of averaged maximum average fitness scores produced using population mutation rates in the range 0.1 to 1 allowing the individual mutation rate to self-optimize starting from rates of 0.1, 0.5, 1 and a random initialization.

Average Fitness Area Scores

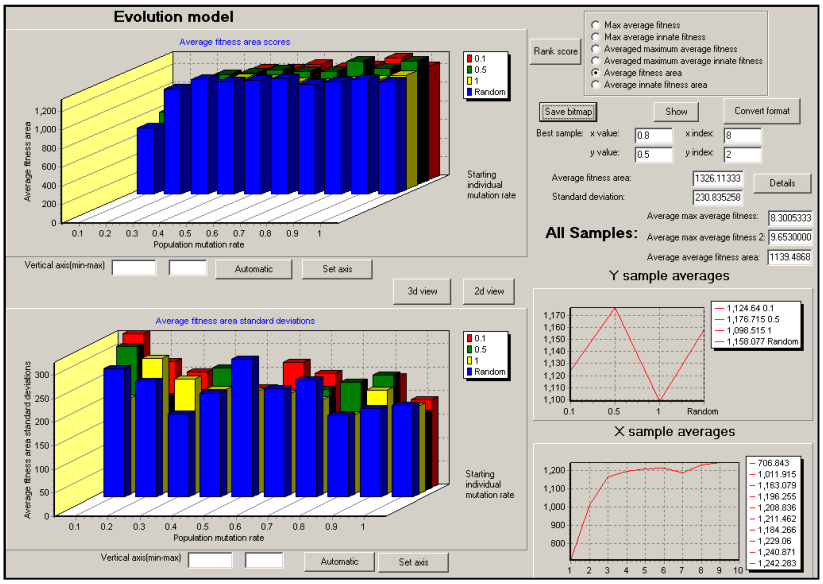


Figure C-5 3d perspective of average fitness area scores produced using population mutation rates in the range 0.1 to 1 allowing the individual mutation rate to self-optimize starting from rates of 0.1, 0.5, 1 and a random initialization.

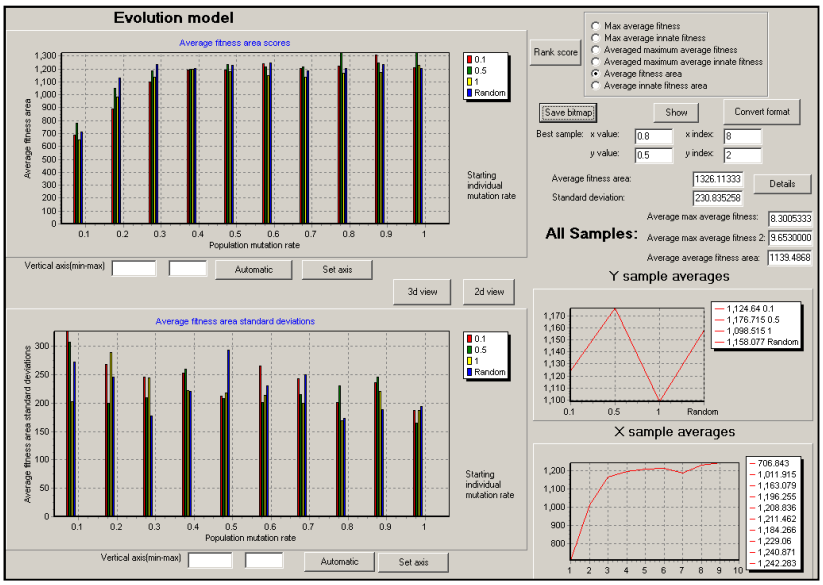


Figure C-6 2d perspective of average fitness area scores produced using population mutation rates in the range 0.1 to 1 allowing the individual mutation rate to self-optimize starting from rates of 0.1, 0.5, 1 and a random initialization.

Appendix D

Evaluation of Our Evolution Model Without Self-optimization in Stable to Highly Unstable Environments

Maximum Average Fitness Scores

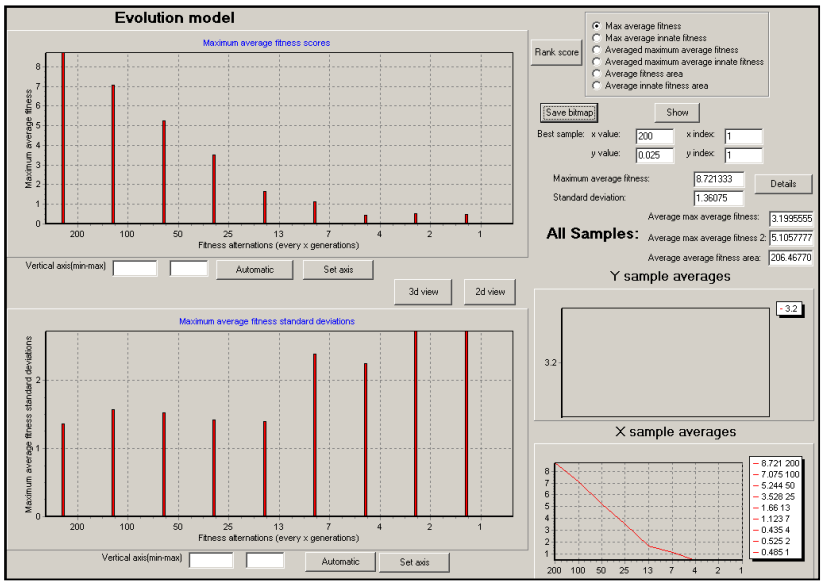


Figure D-1 Maximum average fitness scores produced by our evolution model without self-optimization in stable to highly unstable environments.

Averaged Maximum Average Fitness Scores

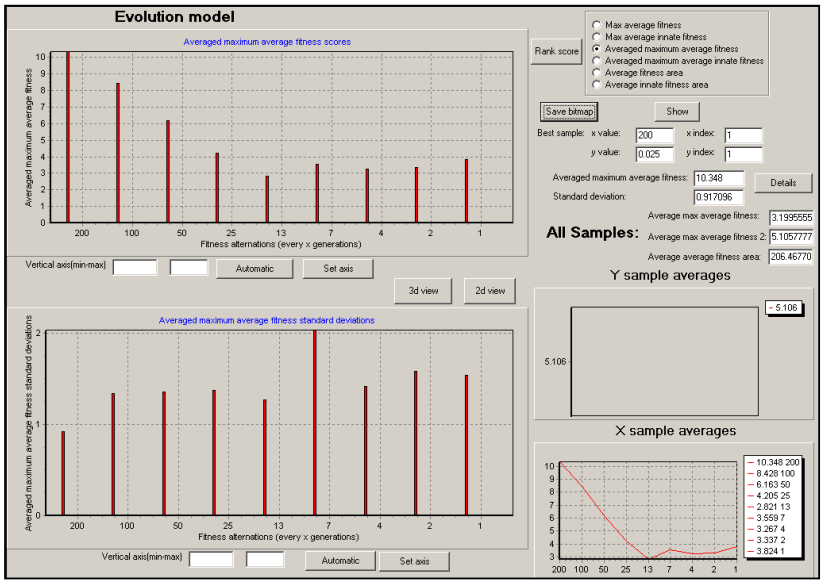


Figure D-2 Averaged maximum average fitness scores produced by our evolution model without self-optimization in stable to highly unstable environments.

Average Fitness Area Scores

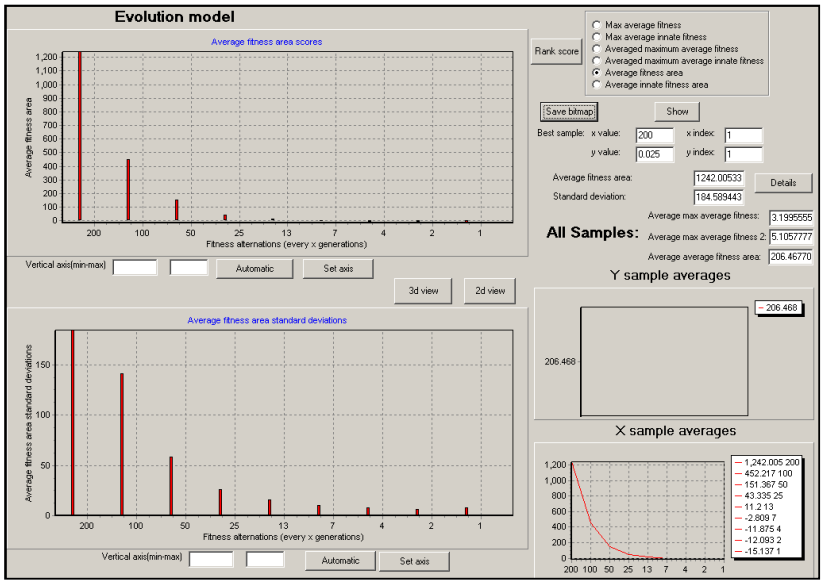


Figure D-3 Average fitness area scores produced by our evolution model without self-optimization in stable to highly unstable environments.

Evaluation of Our Evolution Model With Self-optimization in Stable to Highly Unstable Environments

Maximum Average Fitness Scores

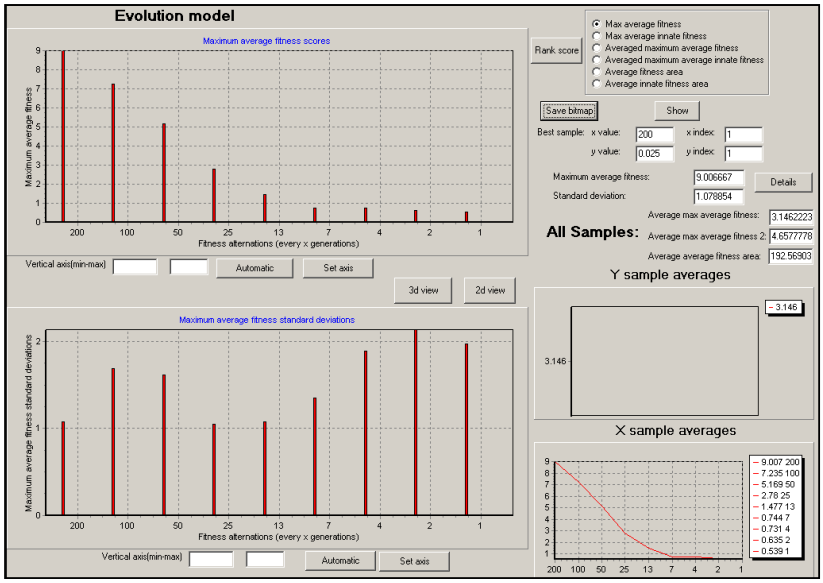


Figure D-4 Maximum average fitness scores produced by our evolution model with self-optimization in stable to highly unstable environments.

Averaged Maximum Average Fitness Scores

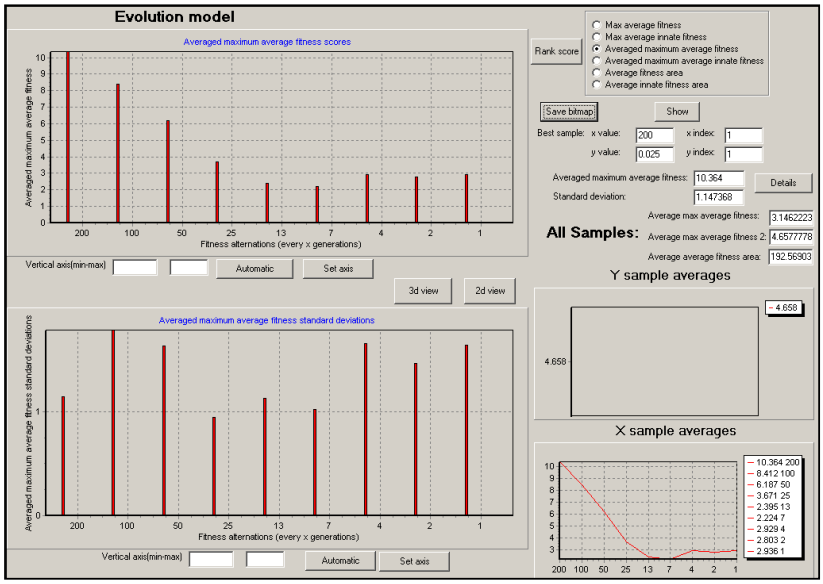


Figure D-5 Averaged maximum average fitness scores produced by our evolution model with self-optimization in stable to highly unstable environments.

Average Fitness Area Scores

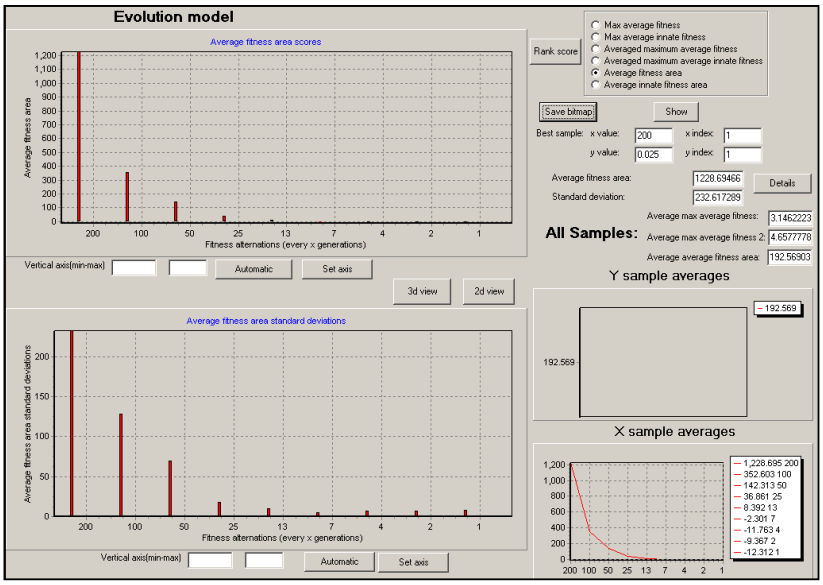


Figure D-6 Average fitness area scores produced by our evolution model with self-optimization in stable to highly unstable environments.

Appendix E

Reinforcement Learning Model

Maximum Average Fitness Scores

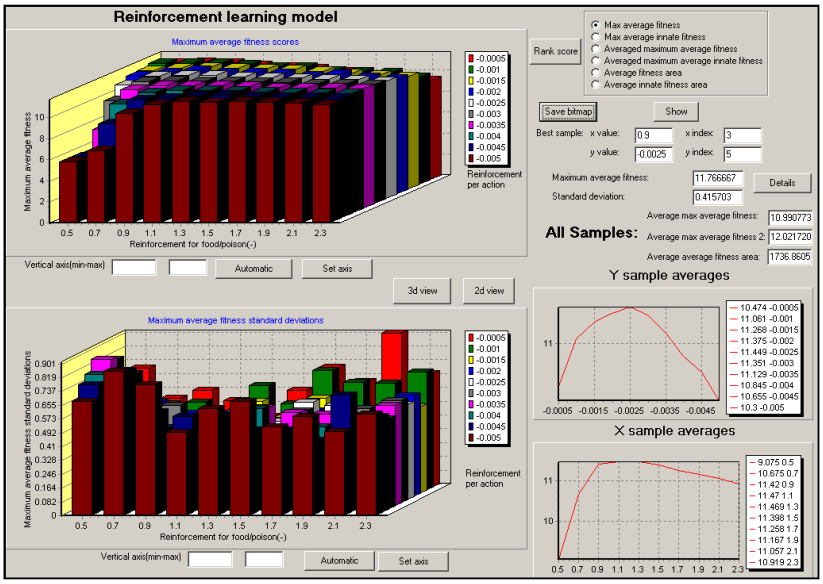


Figure E-1 3d perspective of maximum average fitness scores produced using reinforcement values in the range 0.5 to 2.3 for locating food and –0.5 to –2.3 for locating poison, with reinforcements per action being investigated in the range –0.0005 to –0.005.

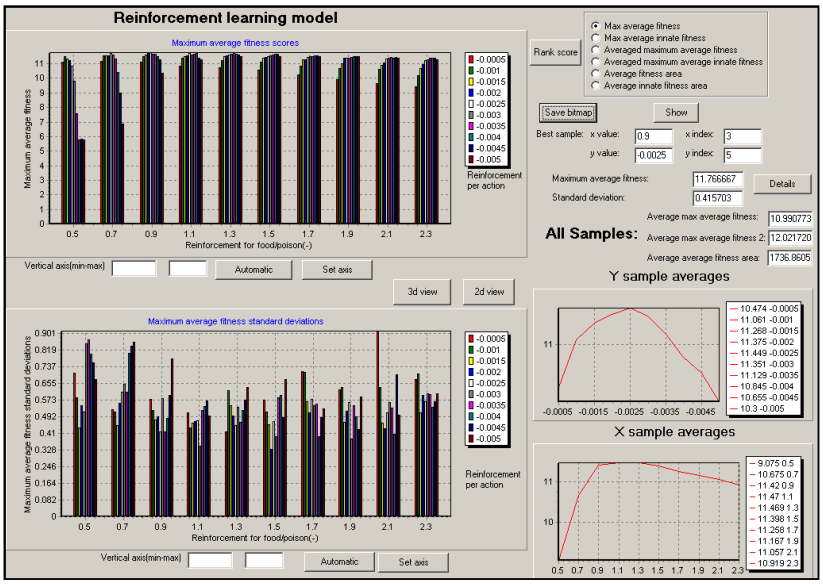


Figure E-2 2d perspective of maximum average fitness scores produced using reinforcement values in the range 0.5 to 2.3 for locating food and –0.5 to –2.3 for locating poison, with reinforcements per action being investigated in the range –0.0005 to –0.005.

Averaged Maximum Average Fitness Scores

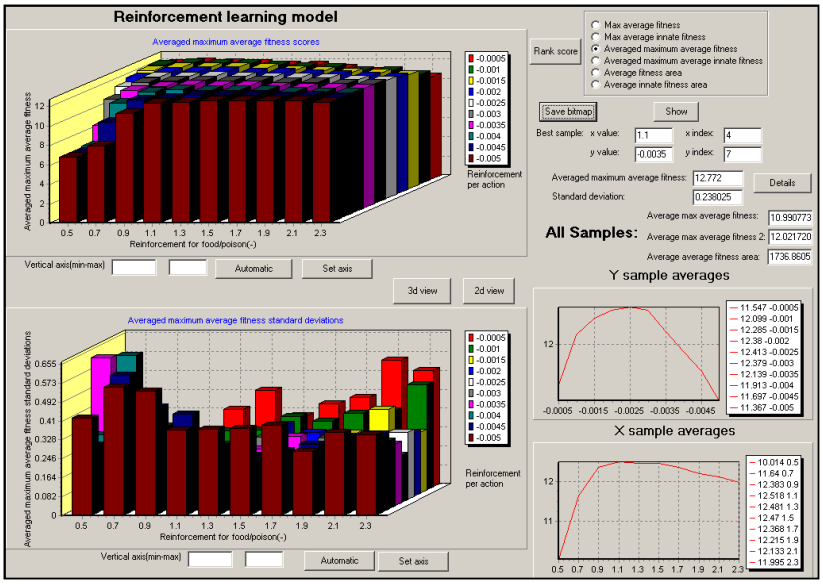


Figure E-3 3d perspective of averaged maximum average fitness scores produced using reinforcement values in the range 0.5 to 2.3 for locating food and –0.5 to –2.3 for locating poison, with reinforcements per action being investigated in the range –0.0005 to –0.005.

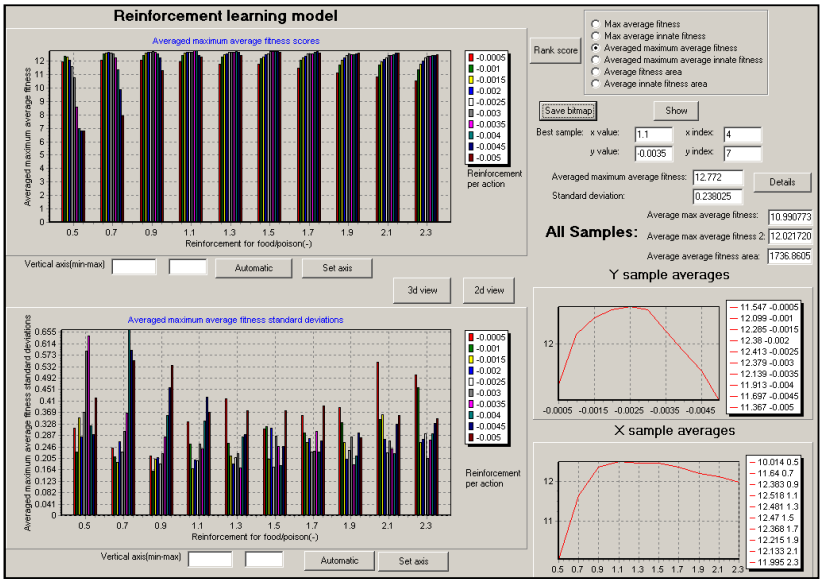


Figure E-4 2d perspective of averaged maximum average fitness scores produced using reinforcement values in the range 0.5 to 2.3 for locating food and –0.5 to –2.3 for locating poison, with reinforcements per action being investigated in the range –0.0005 to –0.005.

Average Fitness Area Scores

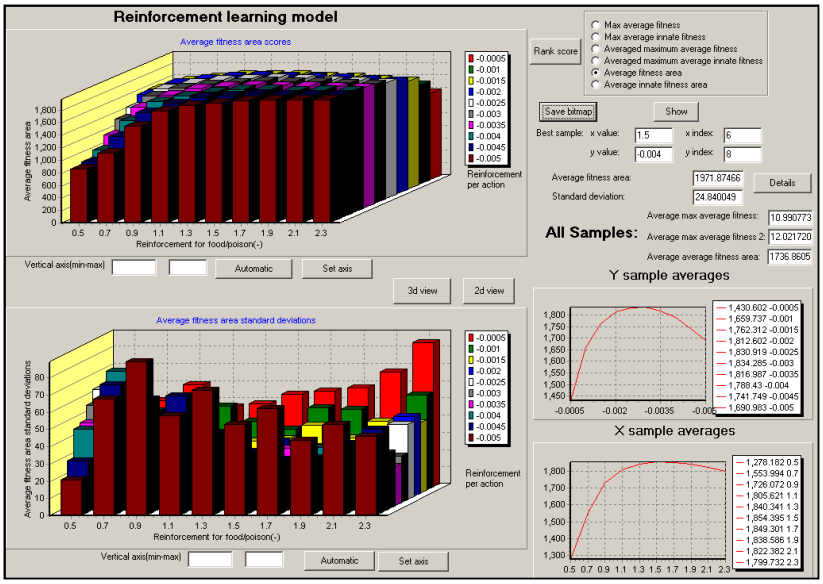


Figure E-5 3d perspective of average fitness area scores produced using reinforcement values in the range 0.5 to 2.3 for locating food and –0.5 to –2.3 for locating poison, with reinforcements per action being investigated in the range –0.0005 to –0.005.

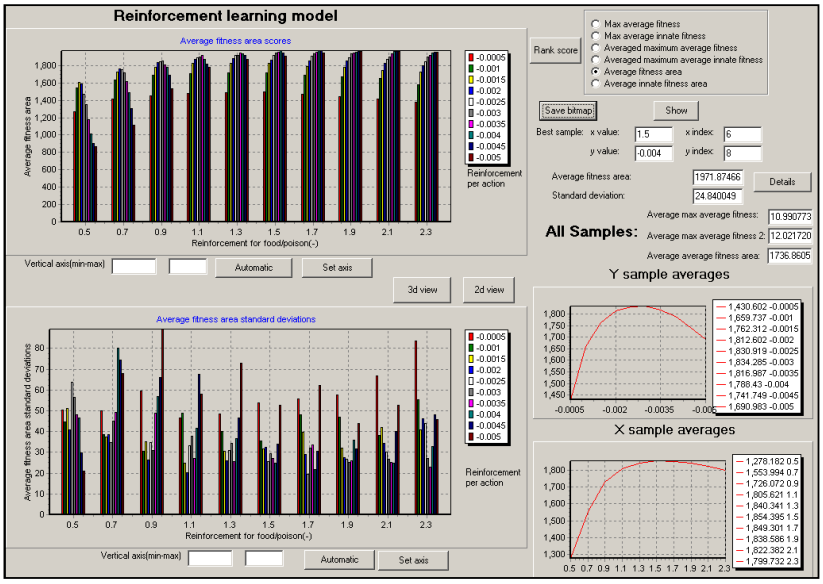


Figure E-6 2d perspective of average fitness area scores produced using reinforcement values in the range 0.5 to 2.3 for locating food and –0.5 to –2.3 for locating poison, with reinforcements per action being investigated in the range –0.0005 to –0.005.

Appendix F

Evaluation of Our Reinforcement Learning Model in Stable to Highly Unstable Environments

Maximum Average Fitness Scores

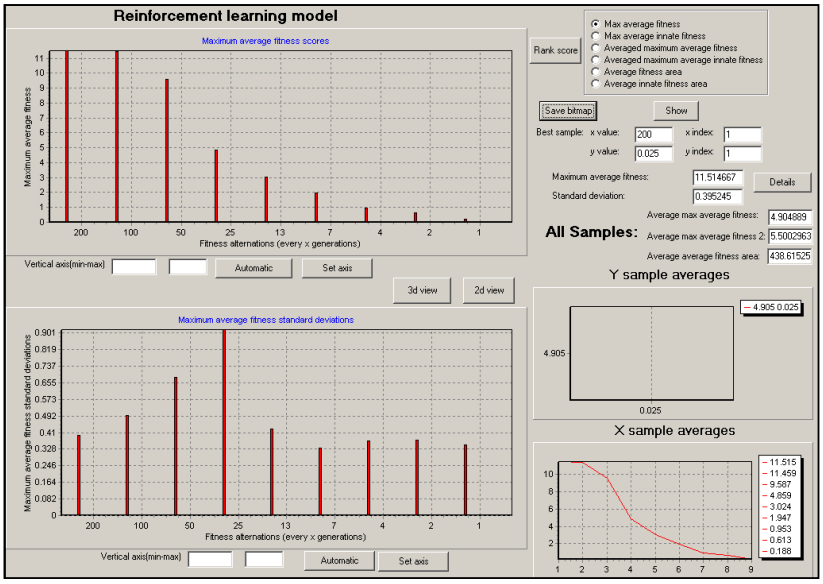


Figure F-1 Maximum average fitness scores produced by our reinforcement learning model in stable to highly unstable environments.

Averaged Maximum Average Fitness Scores

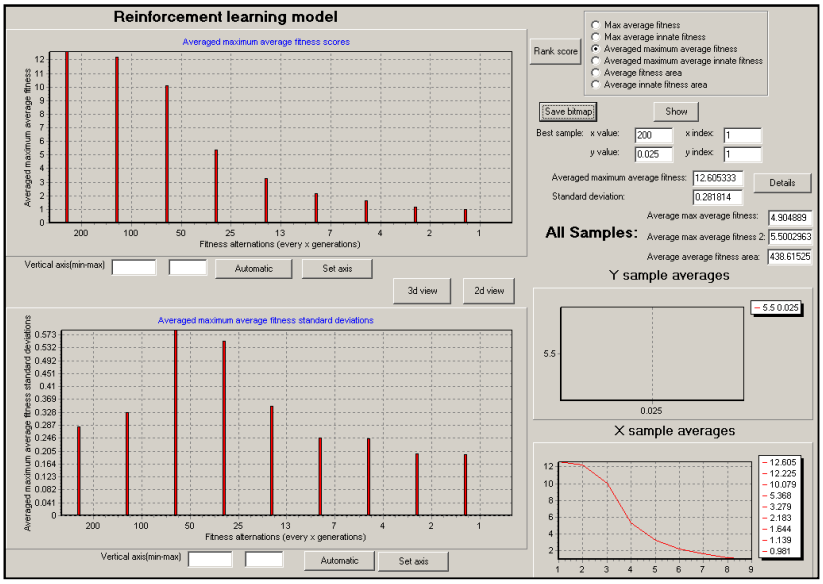


Figure F-2 Averaged maximum average fitness scores produced by our reinforcement learning model in stable to highly unstable environments.

Average Fitness Area Scores

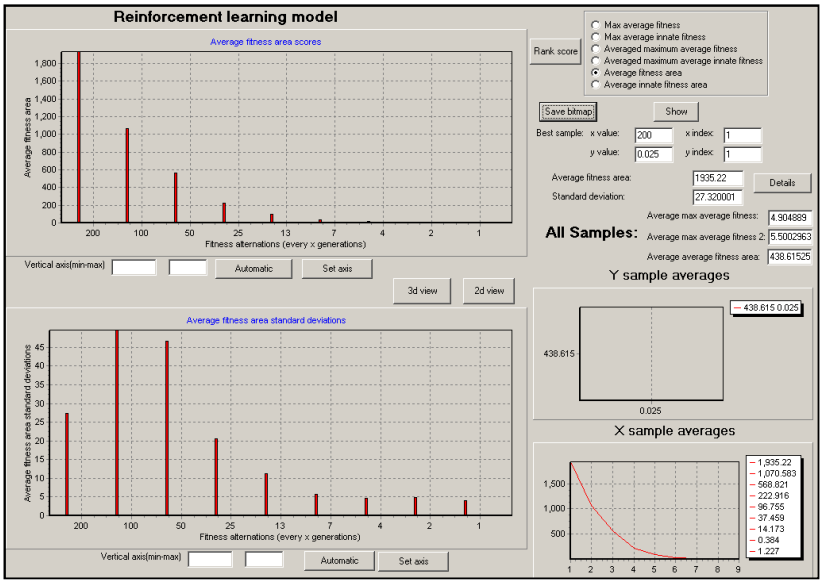


Figure F-3 Average fitness area scores produced by our reinforcement learning model in stable to highly unstable environments.

Appendix G

Darwinian Evolution and Reinforcement Learning Utilizing Different Degrees of Change to Accomplish Feedback Parameter Optimization

Maximum Average Fitness

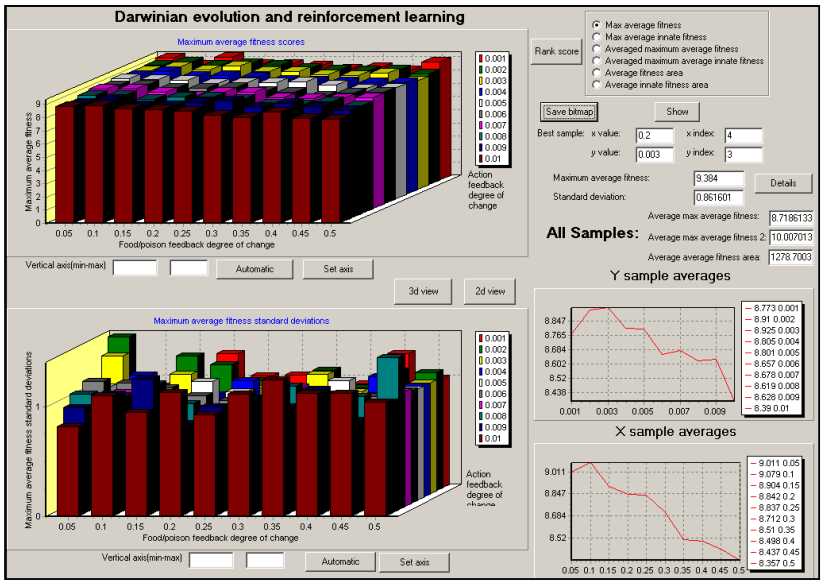


Figure G-1 3d perspective of maximum average fitness scores produced allowing food, poison and action feedback to be adjusted by degrees of change in the range 0.05 to 0.5 for food/poison and 0.001 to 0.01 for feedback per action.

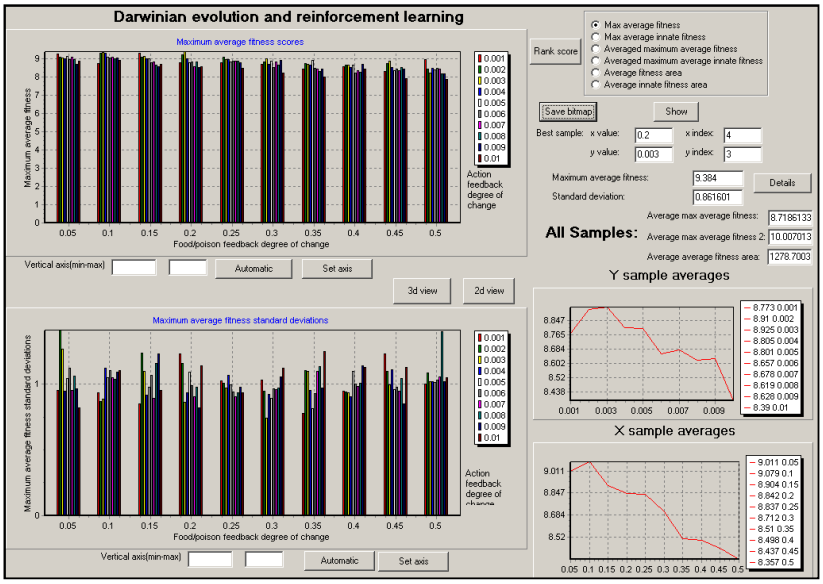


Figure G2 2d perspective of maximum average fitness scores produced allowing food, poison and action feedback to be adjusted by degrees of change in the range 0.05 to 0.5 for food/poison and 0.001 to 0.01 for feedback per action.

Averaged Maximum Average Fitness Scores

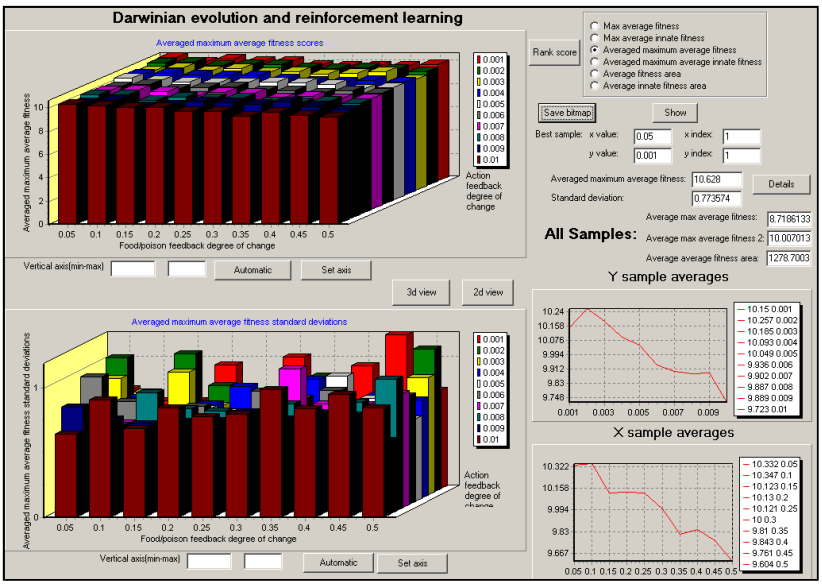


Figure G3 3d perspective of averaged maximum average fitness scores produced allowing food, poison and action feedback to be adjusted by degrees of change in the range 0.05 to 0.5 for food/poison and 0.001 to 0.01 for feedback per action.

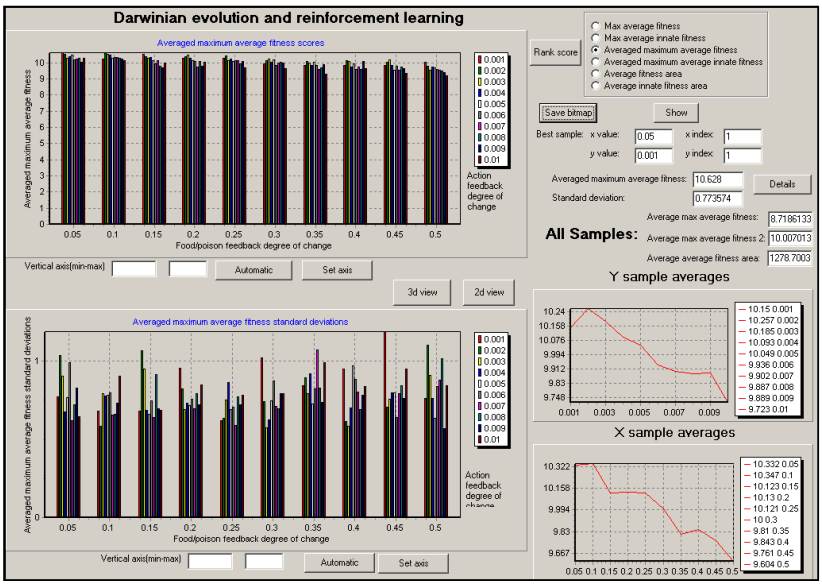


Figure G4 2d perspective of averaged maximum average fitness scores produced allowing food, poison and action feedback to be adjusted by degrees of change in the range 0.05 to 0.5 for food/poison and 0.001 to 0.01 for feedback per action.

Average fitness area scores

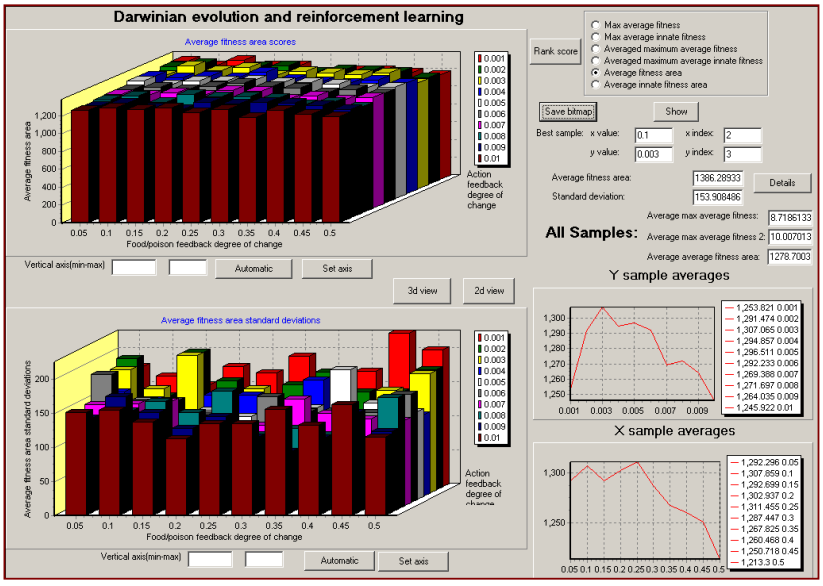


Figure G5 3d perspective of average fitness area scores produced allowing food, poison and action feedback to be adjusted by degrees of change in the range 0.05 to 0.5 for food/poison and 0.001 to 0.01 for feedback per action.

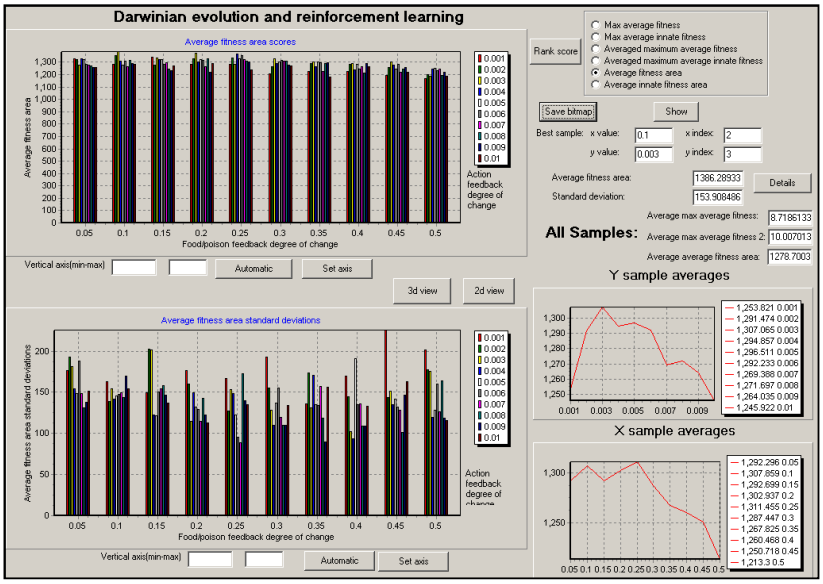


Figure G6 2d perspective of average fitness area scores produced allowing food, poison and action feedback to be adjusted by degrees of change in the range 0.05 to 0.5 for food/poison and 0.001 to 0.01 for feedback per action.

Appendix H

Darwinian Evolution and Reinforcement Learning: Results utilizing a fixed population mutation rate of 0.8 and an individual mutation rate of 0.1 with fixed reinforcements being examined in the range +-0.1 to +-1 for locating food and poison objects and in the range –0.0025 to –0.025 per action.

Maximum Average Fitness Scores

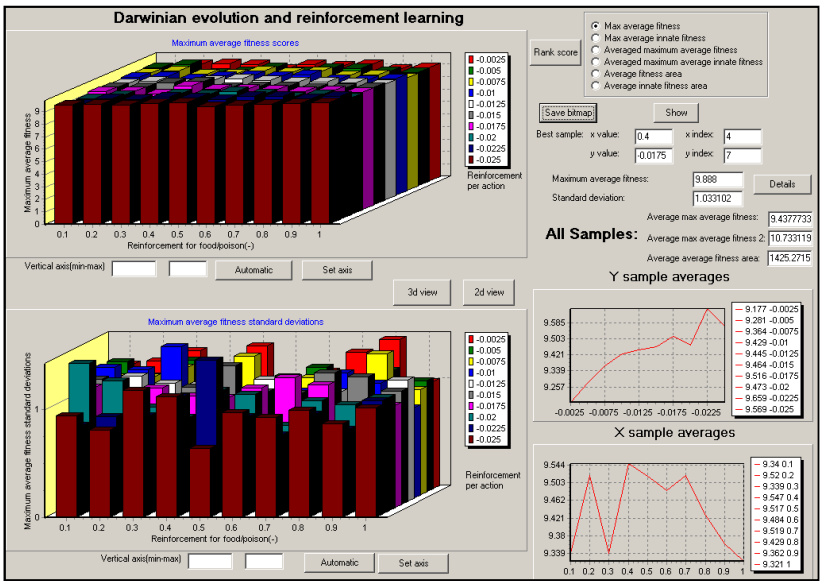


Figure H-1 3d perspective of maximum average fitness scores produced using reinforcement values in the range 0.1 to 1 for locating food and –0.1 to –1 for locating poison, with reinforcements per action being investigated in the range –0.0025 to –0.025.

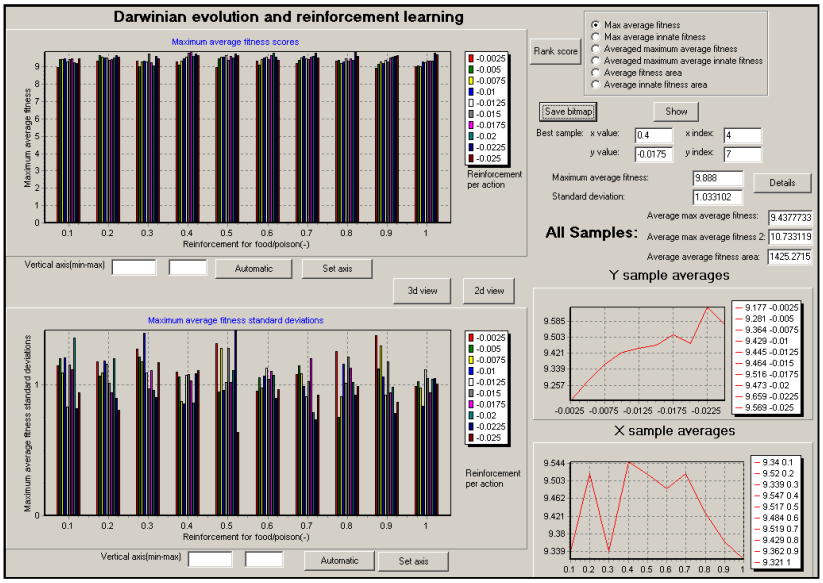


Figure H-2 3d perspective of maximum average fitness scores produced using reinforcement values in the range 0.1 to 1 for locating food and –0.1 to –1 for locating poison, with reinforcements per action being investigated in the range –0.0025 to –0.025.

Averaged Maximum Average Fitness Scores

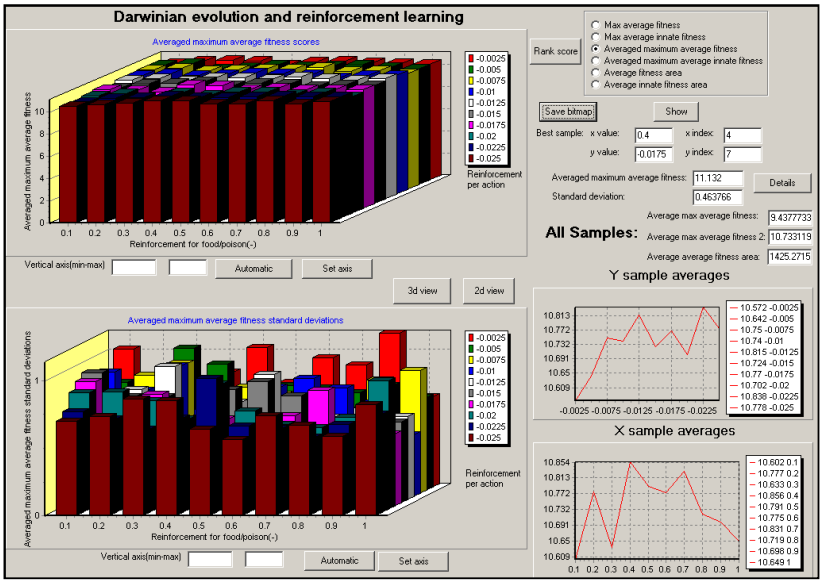


Figure H-3 3d perspective of averaged maximum average fitness scores produced using reinforcement values in the range 0.1 to 1 for locating food and –0.1 to –1 for locating poison, with reinforcements per action being investigated in the range –0.0025 to –0.025.

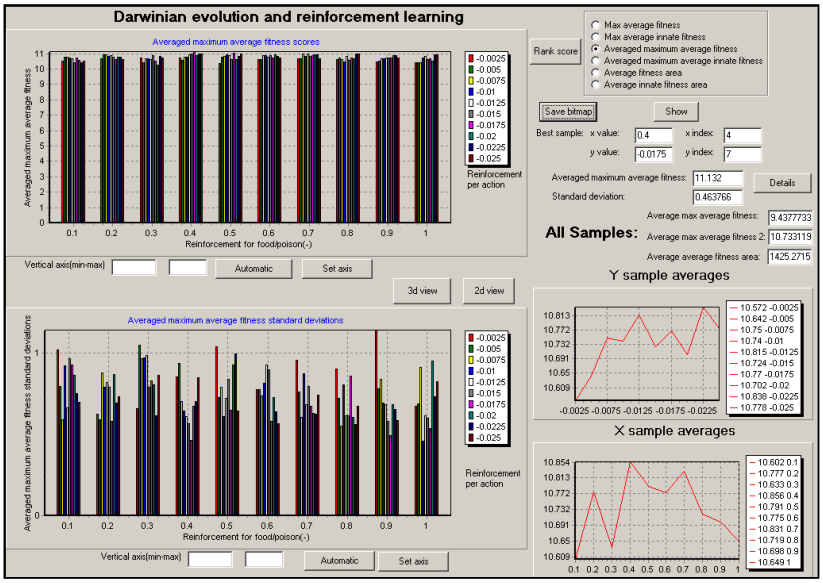


Figure H-4 2d perspective of averaged maximum average fitness scores produced using reinforcement values in the range 0.1 to 1 for locating food and –0.1 to –1 for locating poison, with reinforcements per action being investigated in the range –0.0025 to –0.025.

Average Fitness Area Scores

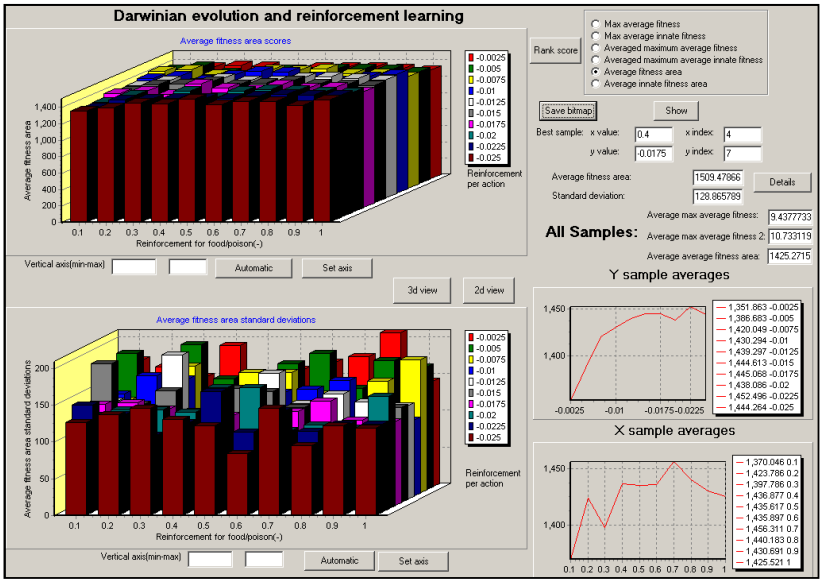


Figure H-5 3d perspective of average fitness area scores produced using reinforcement values in the range 0.1 to 1 for locating food and –0.1 to –1 for locating poison, with reinforcements per action being investigated in the range –0.0025 to –0.025.

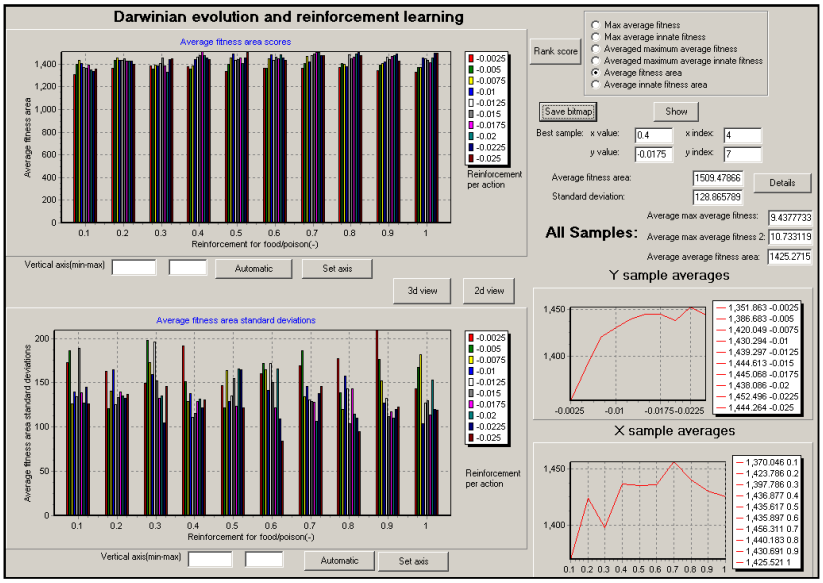


Figure H-6 2d perspective of average fitness area scores produced using reinforcement values in the range 0.1 to 1 for locating food and –0.1 to –1 for locating poison, with reinforcements per action being investigated in the range –0.0025 to –0.025.

Appendix I

Darwinian Evolution and Reinforcement Learning: Results utilizing a fixed population mutation rate of 0.8 and an individual mutation rate that is allowed to self-optimize with fixed reinforcements being examined in the range +-0.1 to +-1 for locating food and poison objects and in the range –0.0025 to –0.025 per action.

Maximum Average Fitness Scores

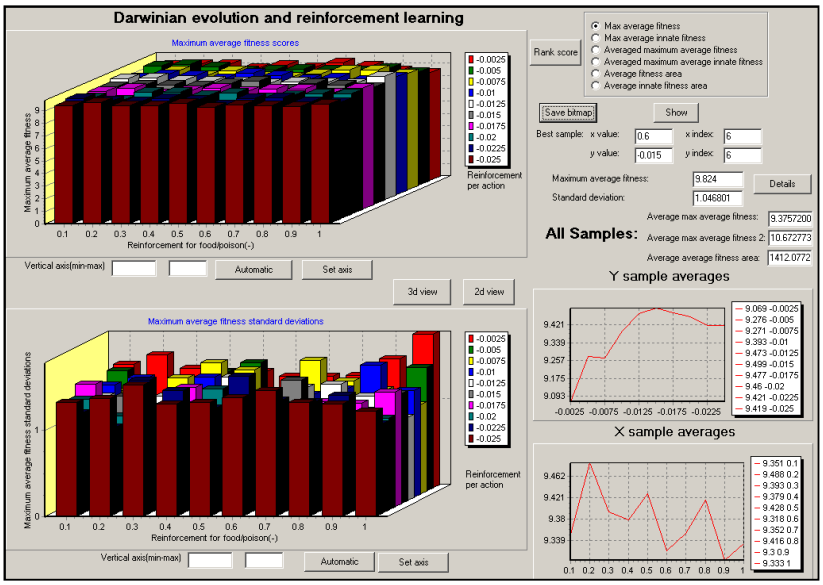


Figure I-1 3d perspective of maximum average fitness scores produced using reinforcement values in the range 0.1 to 1 for locating food and –0.1 to –1 for locating poison, with reinforcements per action being investigated in the range –0.0025 to –0.025.

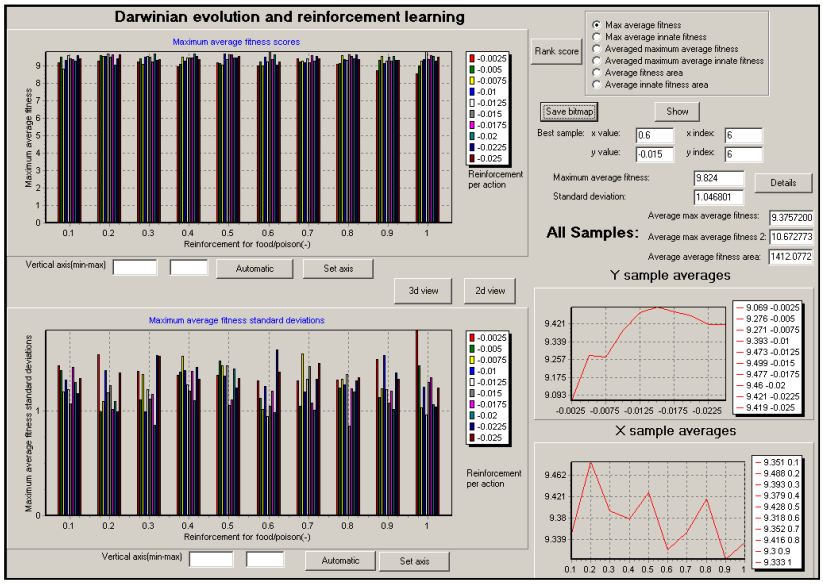


Figure I-2 2d perspective of maximum average fitness scores produced using reinforcement values in the range 0.1 to 1 for locating food and –0.1 to –1 for locating poison, with reinforcements per action being investigated in the range –0.0025 to –0.025.

Averaged Maximum Average Fitness Scores

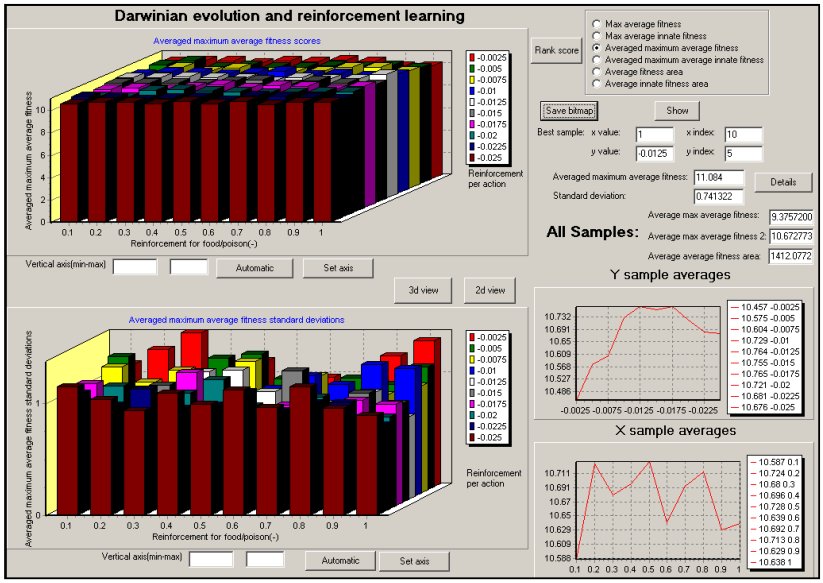


Figure I-3 3d perspective of averaged maximum average fitness scores produced using reinforcement values in the range 0.1 to 1 for locating food and –0.1 to –1 for locating poison, with reinforcements per action being investigated in the range –0.0025 to –0.025.

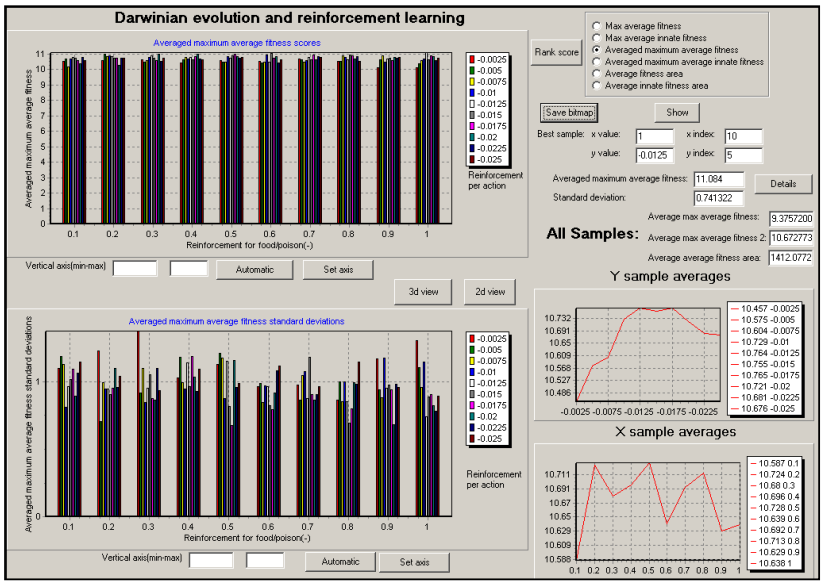


Figure I-4 2d perspective of averaged maximum average fitness scores produced using reinforcement values in the range 0.1 to 1 for locating food and –0.1 to –1 for locating poison, with reinforcements per action being investigated in the range –0.0025 to –0.025.

Average Fitness Area Scores

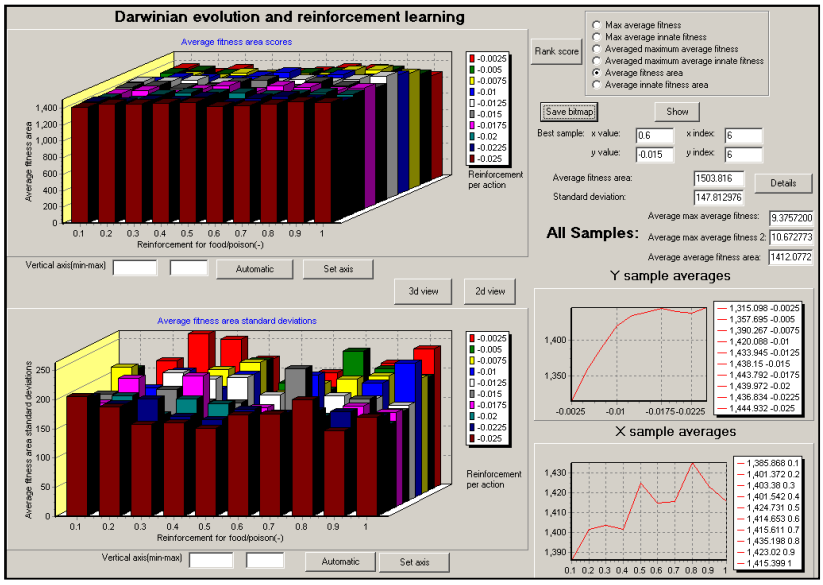


Figure I-5 3d perspective of average fitness area scores produced using reinforcement values in the range 0.1 to 1 for locating food and –0.1 to –1 for locating poison, with reinforcements per action being investigated in the range –0.0025 to –0.025.

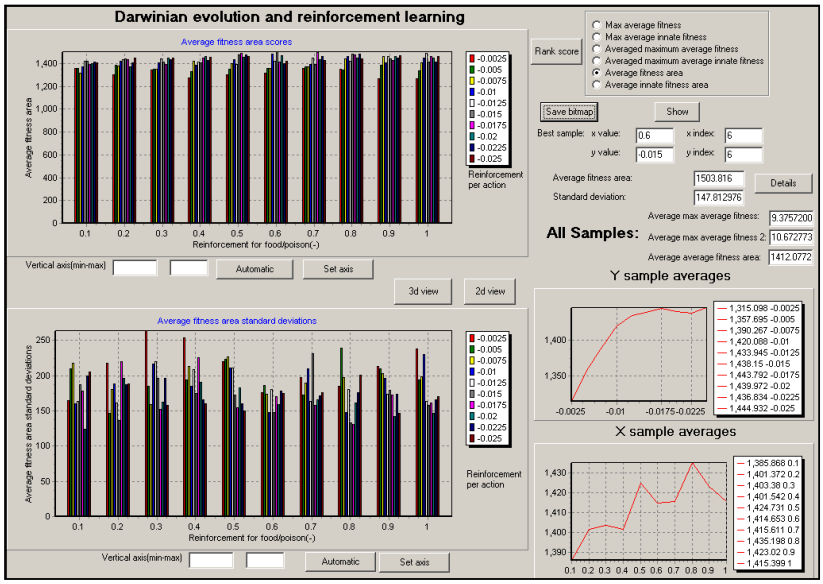


Figure I-6 2d perspective of average fitness area scores produced using reinforcement values in the range 0.1 to 1 for locating food and –0.1 to –1 for locating poison, with reinforcements per action being investigated in the range –0.0025 to –0.025.

Appendix J

Evaluation of our Darwinian Evolution and Reinforcement Learning Model With Fixed Mutation Rates and Optimizing Reinforcement Feedback Values in Stable to Highly Unstable Environments

The following data was produced using a population mutation rate of 0.8 and an individual mutation rate of 0.1 with reinforcement feedback values being optimized starting from 0.4 for locating food, -0.4 for locating poison and –0.02 per action.

Maximum Average Fitness Scores

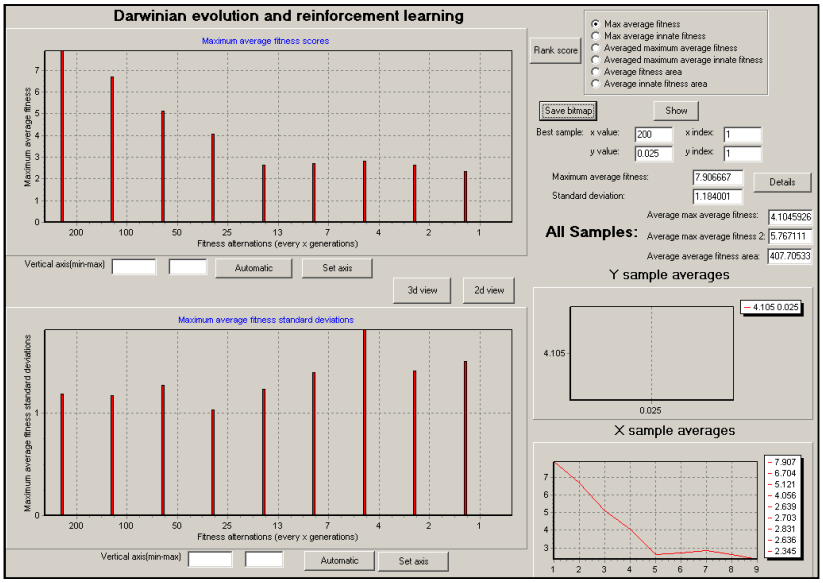


Figure J-1 Maximum average fitness scores produced by our Darwinian evolution and reinforcement learning model in stable to highly unstable environments utilizing fixed mutation rates and allowing reinforcement feedback values to be optimized.

Averaged Maximum Average Fitness Scores

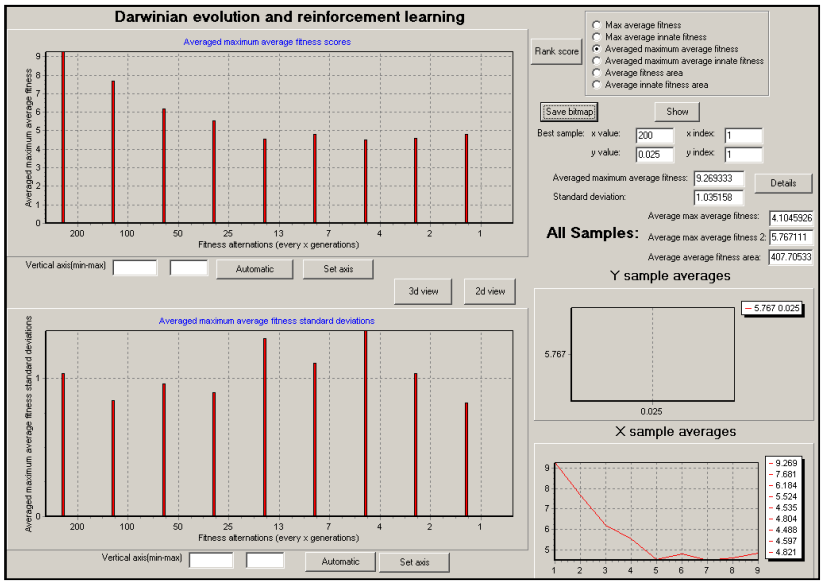


Figure J-2 Averaged maximum average fitness scores produced by our Darwinian evolution and reinforcement learning model in stable to highly unstable environments utilizing fixed mutation rates and allowing reinforcement feedback values to be optimized.

Average Fitness Area Scores

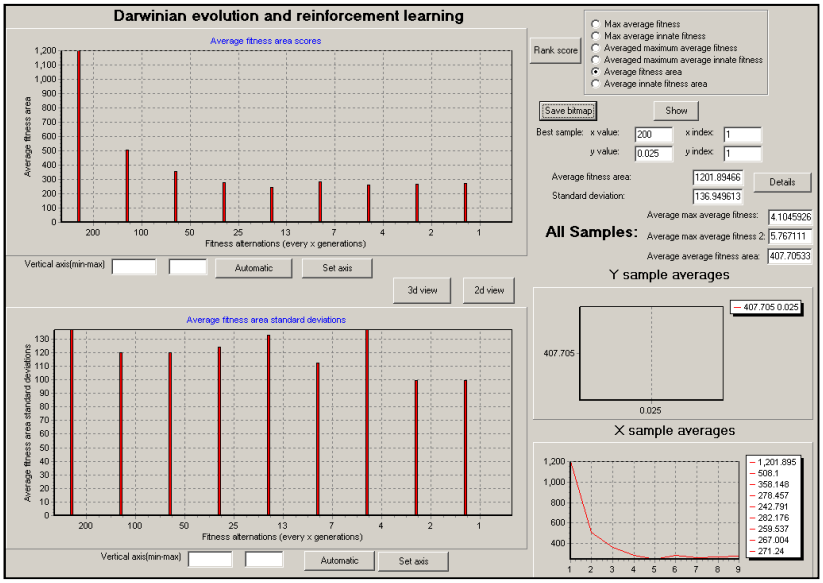


Figure J-3 Average fitness area scores produced by our Darwinian evolution and reinforcement learning model in stable to highly unstable environments utilizing fixed mutation rates and allowing reinforcement feedback values to be optimized.

Evaluation of our Darwinian Evolution and Reinforcement Learning Model Optimizing the Individual's Mutation Rate and Optimizing Reinforcement Feedback Values in Stable to Highly Unstable Environments

The following data was produced using a population mutation rate of 0.8 and allowing the individual's mutation rate to be optimized with reinforcement feedback values being optimized starting from 0.4 for locating food, -0.4 for locating poison and –0.02 per action.

Maximum Average Fitness Scores

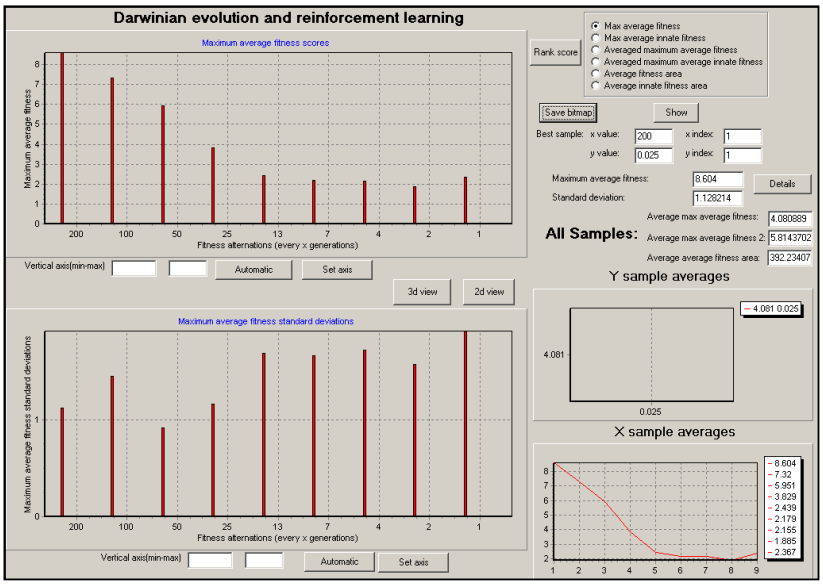


Figure J-4 Maximum average fitness scores produced by our Darwinian evolution and reinforcement learning model in stable to highly unstable environments allowing the individual's mutation rate and reinforcement feedback values to be optimized.

Averaged Maximum Average Fitness Scores

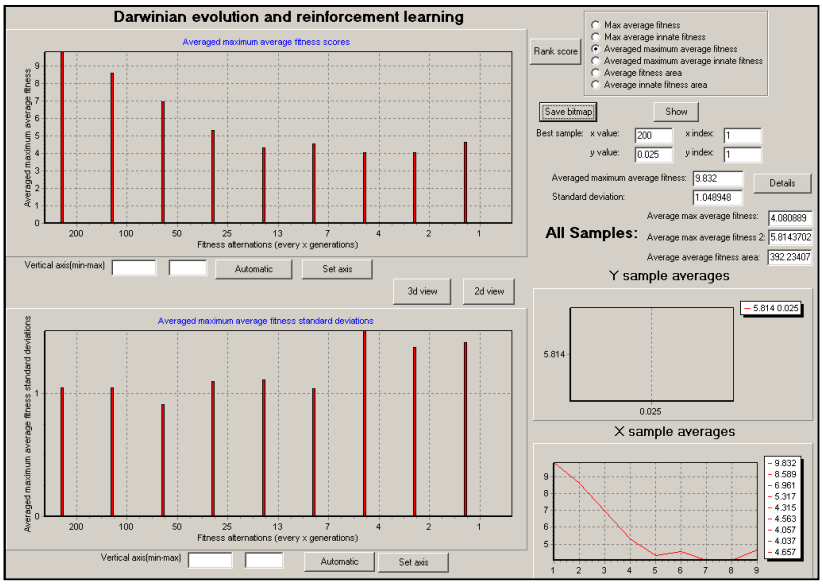


Figure J-5 Averaged maximum average fitness scores produced by our Darwinian evolution and reinforcement learning model in stable to highly unstable environments allowing the individual's mutation rate and reinforcement feedback values to be optimized.

Average Fitness Area Scores

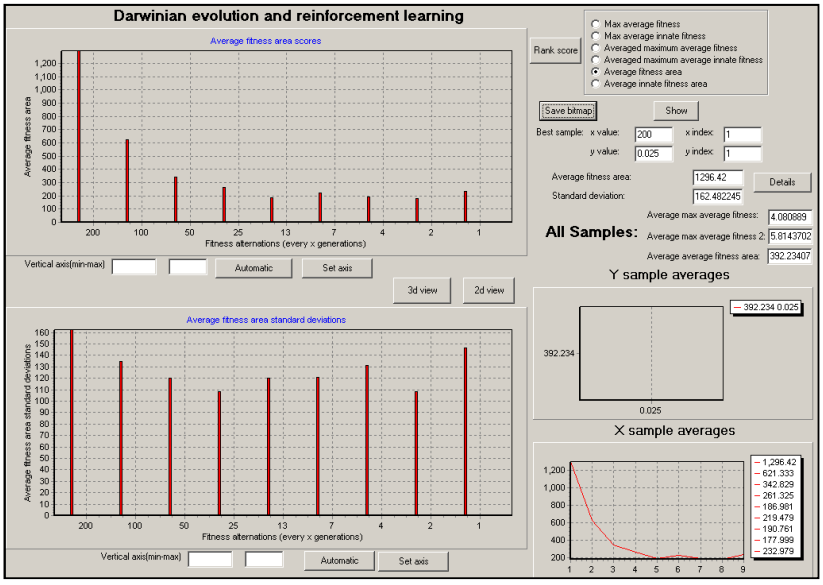


Figure J-6 Average fitness area scores produced by our Darwinian evolution and reinforcement learning model in stable to highly unstable environments allowing the individual's mutation rate and reinforcement feedback values to be optimized.

Evaluation of our Darwinian Evolution and Reinforcement Learning Model Using Fixed Mutation Rates and Fixed Reinforcement Feedback Values in Stable to Highly Unstable Environments

The following data was produced using a population mutation rate of 0.8 and an individual mutation rate of 0.1 with reinforcement feedback values being fixed at 0.4 for locating food, -0.4 for locating poison and –0.02 per action.

Maximum Average Fitness Scores

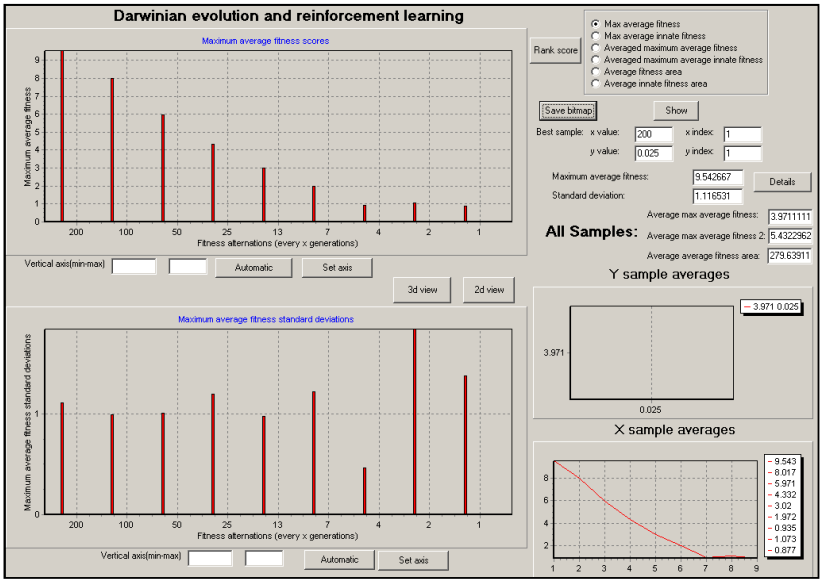


Figure J-7 Maximum average fitness scores produced by our Darwinian evolution and reinforcement learning model in stable to highly unstable environments using fixed mutation rates and fixed reinforcement feedback values.

Averaged Maximum Average Fitness Scores

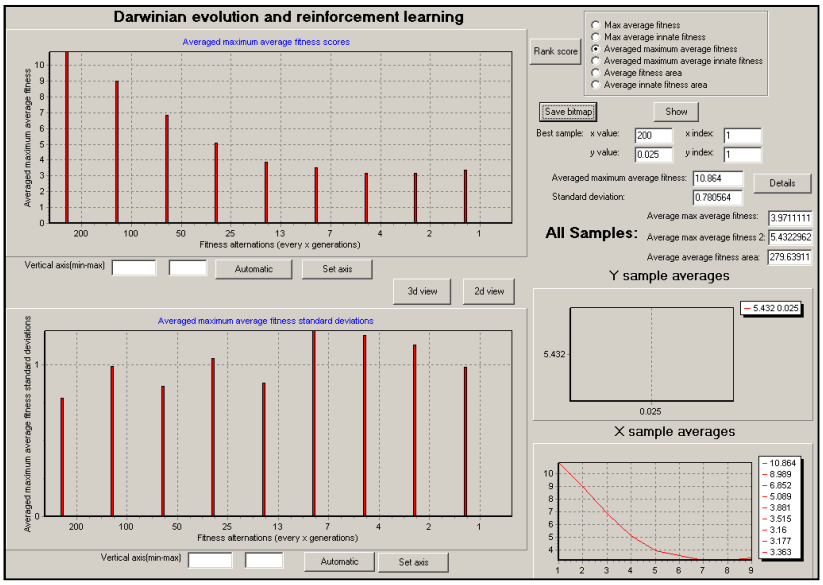


Figure J-8 Averaged maximum average fitness scores produced by our Darwinian evolution and reinforcement learning model in stable to highly unstable environments using fixed mutation rates and fixed reinforcement feedback values.

Average Fitness Area Scores

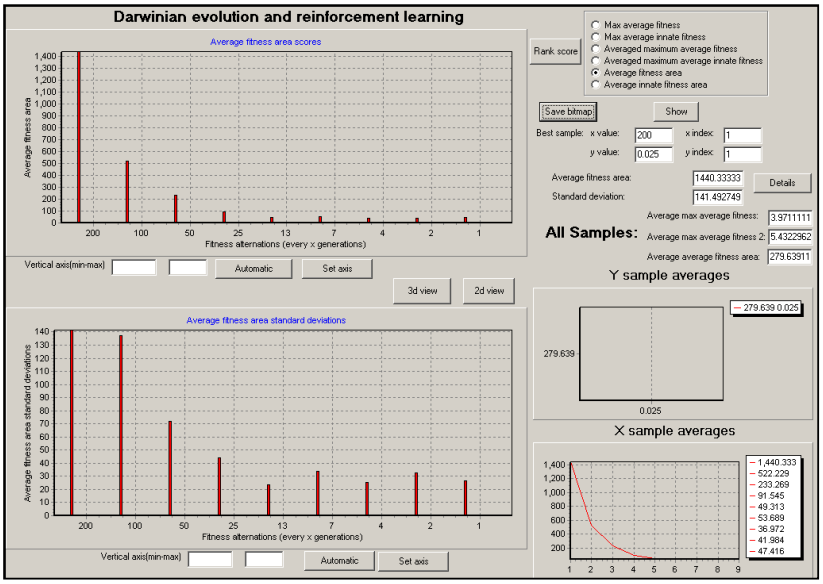


Figure J-9 Average fitness area scores produced by our Darwinian evolution and reinforcement learning model in stable to highly unstable environments using fixed mutation rates and fixed reinforcement feedback values.

Evaluation of our Darwinian Evolution and Reinforcement Learning Model Allowing the Individual's Mutation Rate to Optimize and Using Fixed Reinforcement Feedback Values in Stable to Highly Unstable Environments

The following data was produced using a population mutation rate of 0.8 and allowing the individual's mutation rate to optimize with reinforcement feedback values being fixed at 0.4 for locating food, -0.4 for locating poison and –0.02 per action.

Maximum Average Fitness Scores

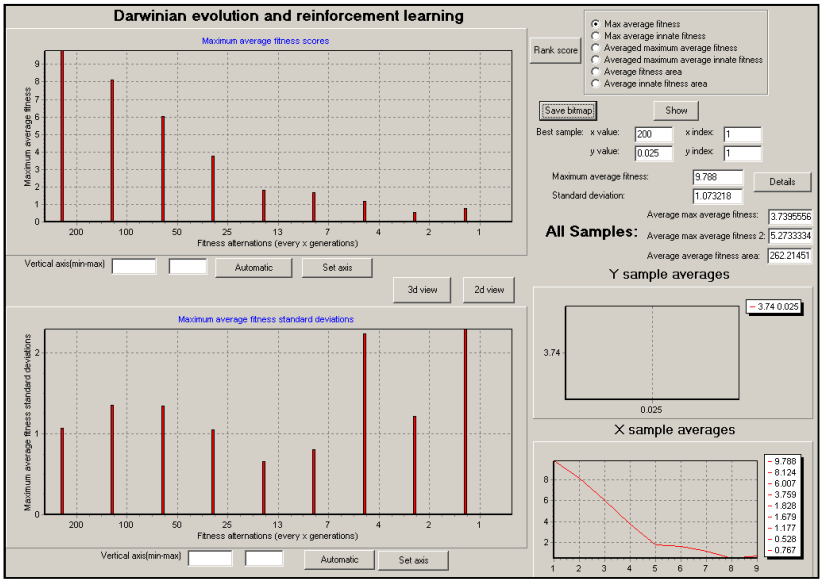


Figure J-10 Maximum average fitness scores produced by our Darwinian evolution and reinforcement learning model in stable to highly unstable environments allowing the individual's mutation rate to be optimized and using fixed reinforcement feedback values.

Averaged Maximum Average Fitness Scores

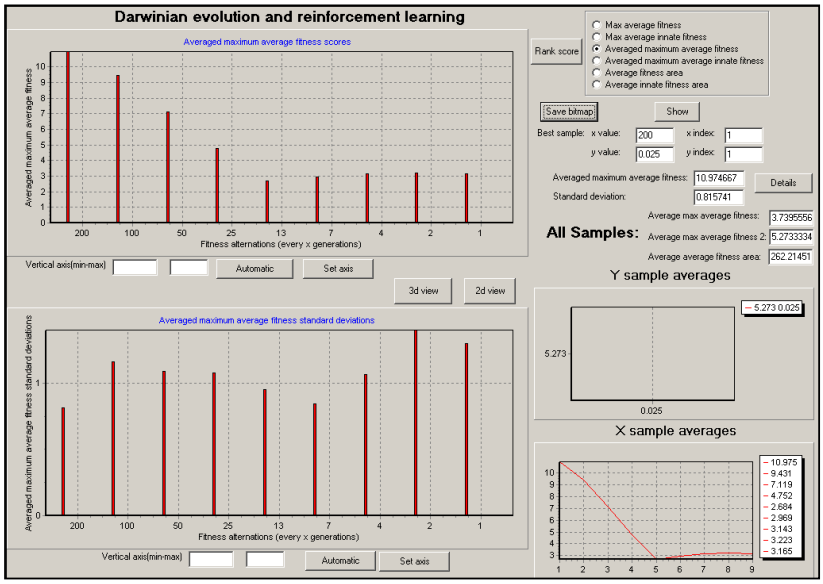


Figure J-11 Averaged maximum average fitness scores produced by our Darwinian evolution and reinforcement learning model in stable to highly unstable environments allowing the individual's mutation rate to be optimized and using fixed reinforcement feedback values.

Average Fitness Area Scores

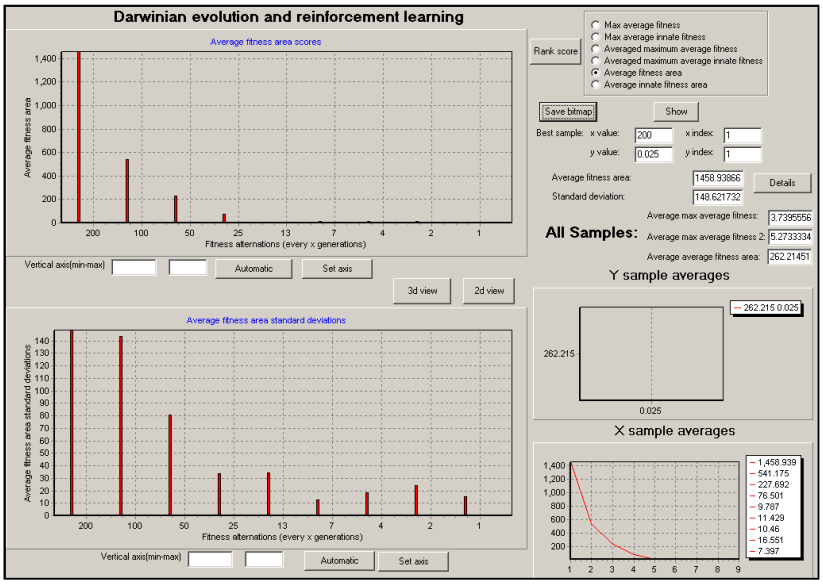


Figure J-12 Average fitness area scores produced by our Darwinian evolution and reinforcement learning model in stable to highly unstable environments allowing the individual's mutation rate to be optimized and using fixed reinforcement feedback values.

Appendix K

Lamarckian Evolution and Reinforcement Learning: Results utilizing no mutations and fixed reinforcements in the range +-1 to +-10 for locating food and poison objects and in the range –0.012 to –0.021 per action.

Maximum Average Fitness Scores

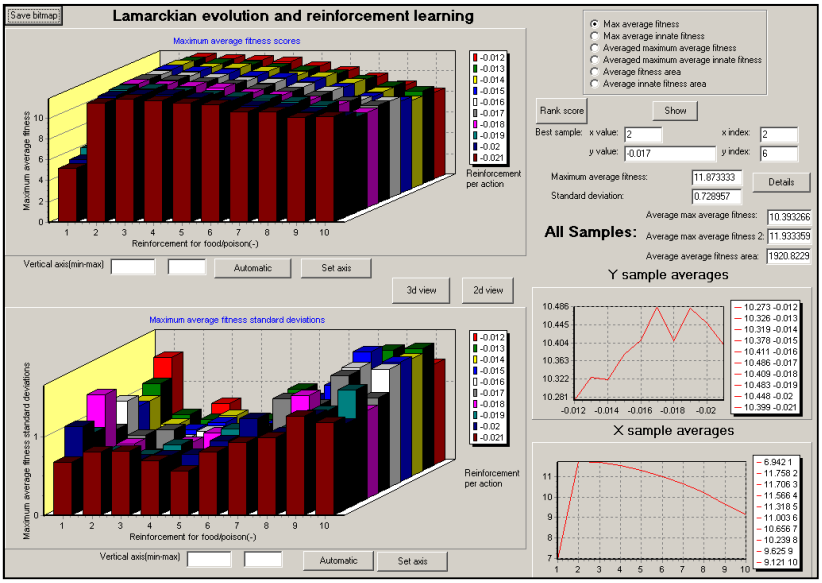


Figure K-1 3d perspective of maximum average fitness scores produced using reinforcement values in the range 1 to 10 for locating food and –1 to –10 for locating poison, with reinforcements per action being investigated in the range –0.012 to –0.021.

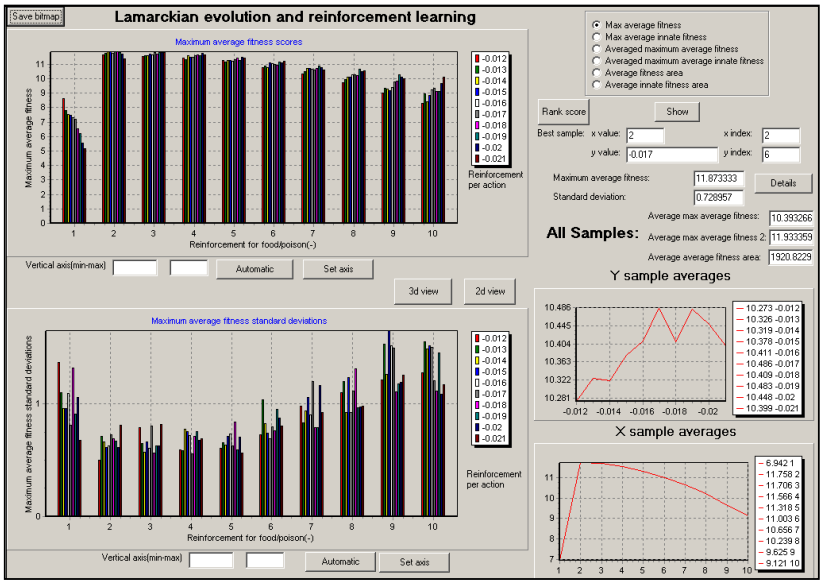


Figure K-2 2d perspective of maximum average fitness scores produced using reinforcement values in the range 1 to 10 for locating food and –1 to –10 for locating poison, with reinforcements per action being investigated in the range –0.012 to –0.021.

Averaged Maximum Average Fitness Scores

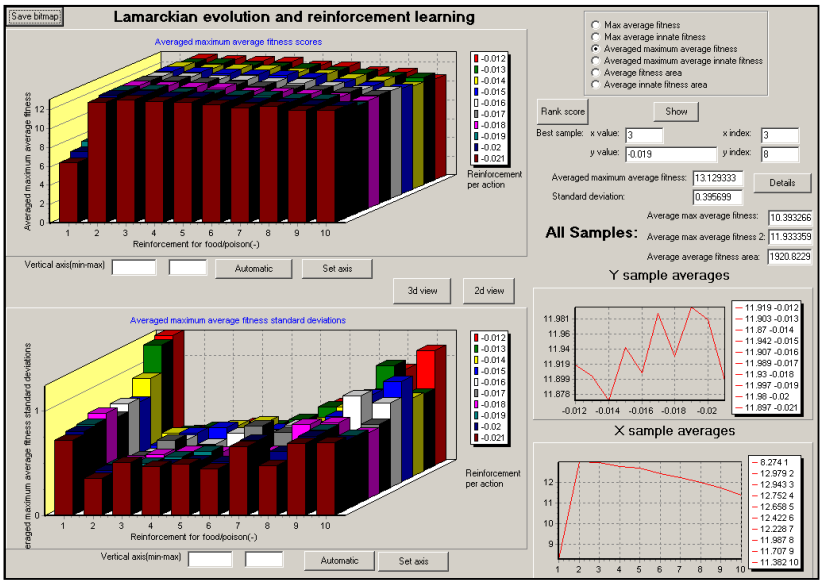


Figure K-3 3d perspective of averaged maximum average fitness scores produced using reinforcement values in the range 1 to 10 for locating food and –1 to –10 for locating poison, with reinforcements per action being investigated in the range –0.012 to –0.021.

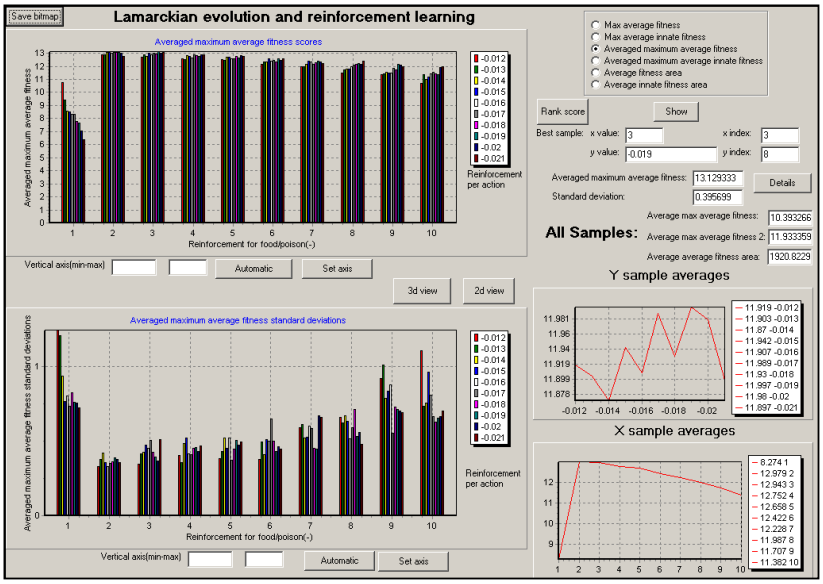


Figure K-4 2d perspective of maximum average fitness scores produced using reinforcement values in the range 1 to 10 for locating food and –1 to –10 for locating poison, with reinforcements per action being investigated in the range –0.012 to –0.021.

Average Fitness Area Scores

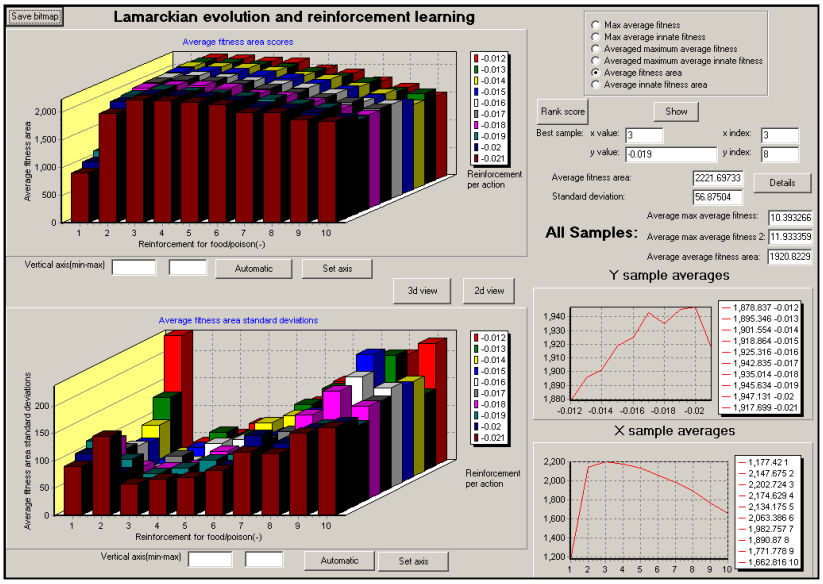


Figure K-5 3d perspective of average fitness area scores produced using reinforcement values in the range 1 to 10 for locating food and –1 to –10 for locating poison, with reinforcements per action being investigated in the range –0.012 to –0.021.

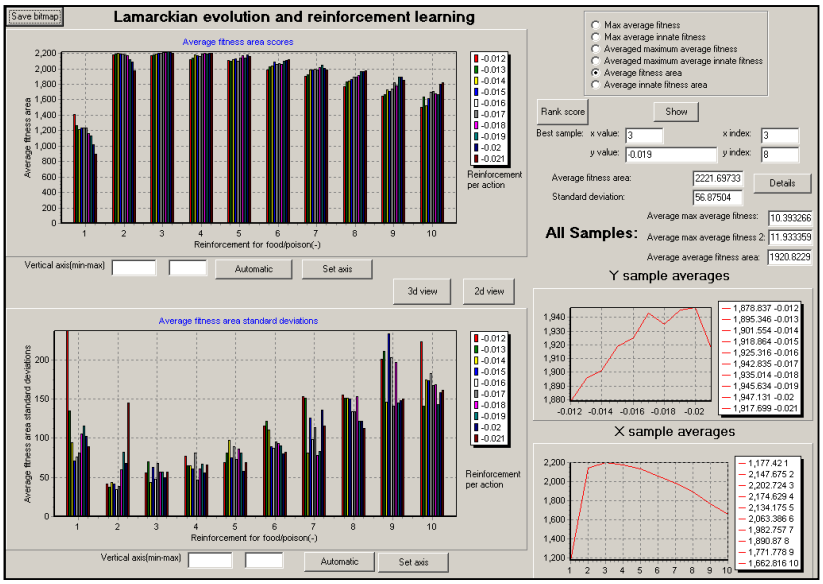


Figure K-6 2d perspective of average fitness area scores produced using reinforcement values in the range 1 to 10 for locating food and –1 to –10 for locating poison, with reinforcements per action being investigated in the range –0.012 to –0.021.

Appendix L

Lamarckian Evolution and Reinforcement Learning Utilizing Different Degrees of Change to Accomplish Feedback Parameter Optimization

Maximum Average Fitness Scores

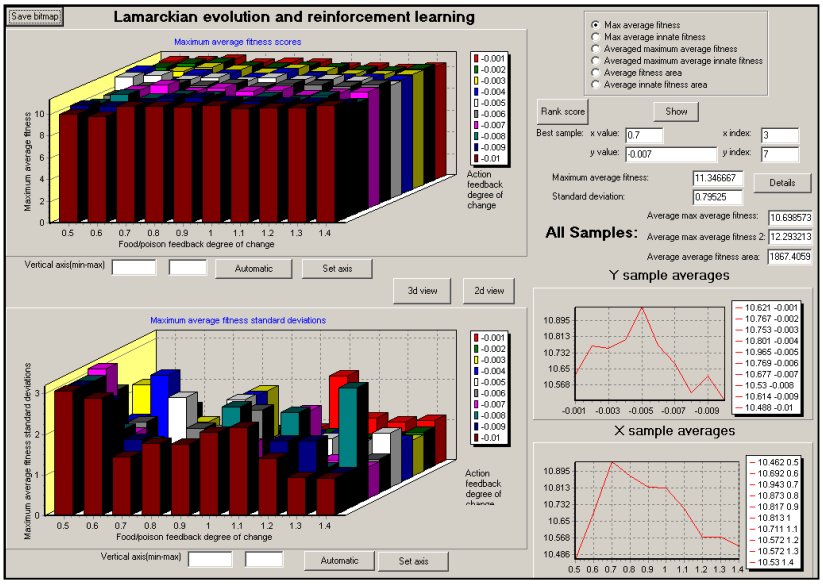


Figure L-1 3d perspective of maximum average fitness scores produced allowing food, poison and action feedback to be adjusted by degrees of change in the range 0.5 to 1.4 for food/poison and 0.001 to 0.01 for feedback per action.

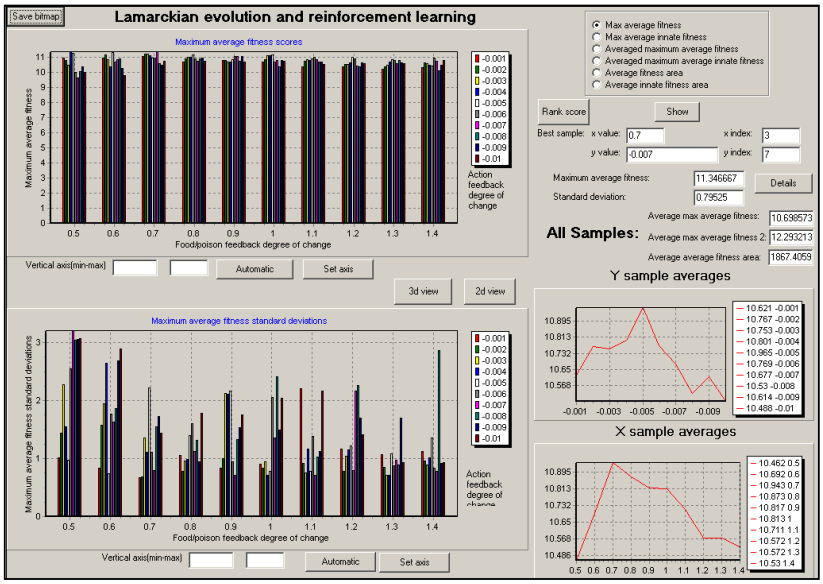


Figure L-2 2d perspective of maximum average fitness scores produced allowing food, poison and action feedback to be adjusted by degrees of change in the range 0.5 to 1.4 for food/poison and 0.001 to 0.01 for feedback per action.

Averaged Maximum Average Fitness Scores

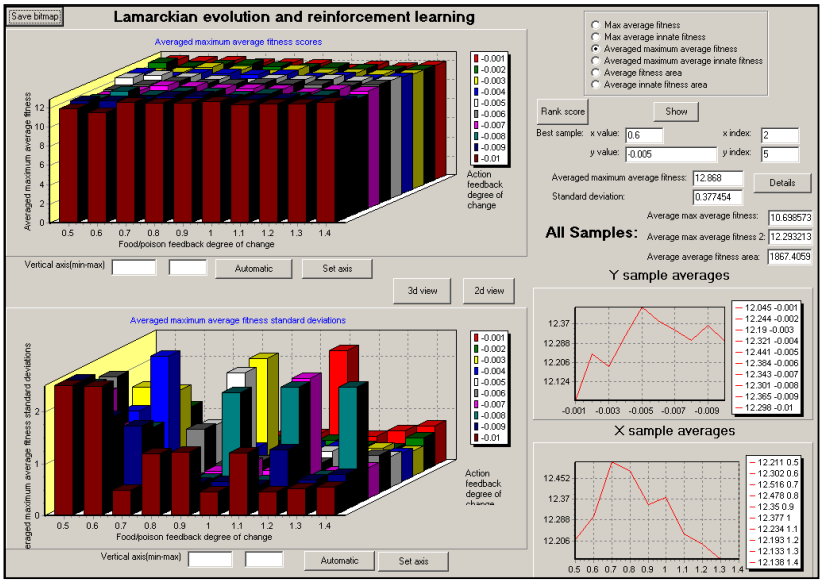


Figure L-3 3d perspective of averaged maximum average fitness scores produced allowing food, poison and action feedback to be adjusted by degrees of change in the range 0.5 to 1.4 for food/poison and 0.001 to 0.01 for feedback per action.

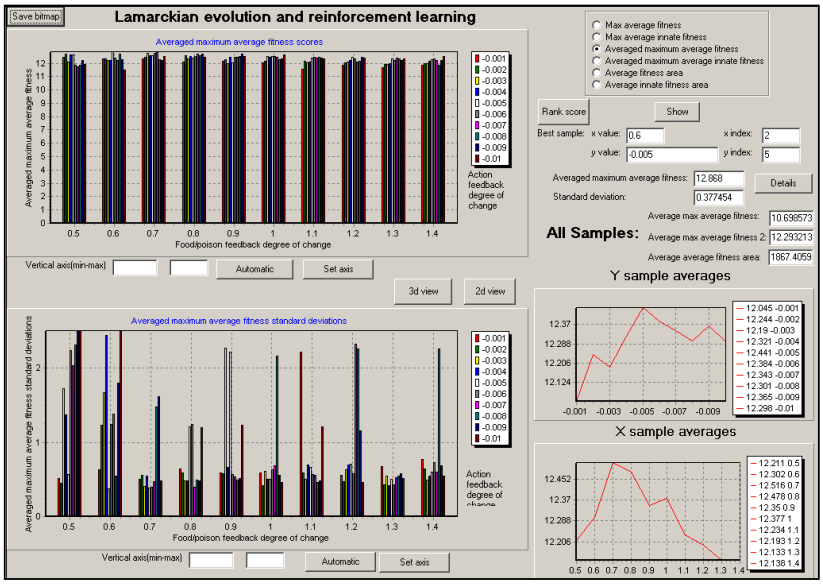


Figure L-4 2d perspective of averaged maximum average fitness scores produced allowing food, poison and action feedback to be adjusted by degrees of change in the range 0.5 to 1.4 for food/poison and 0.001 to 0.01 for feedback per action.

Average Fitness Area Scores

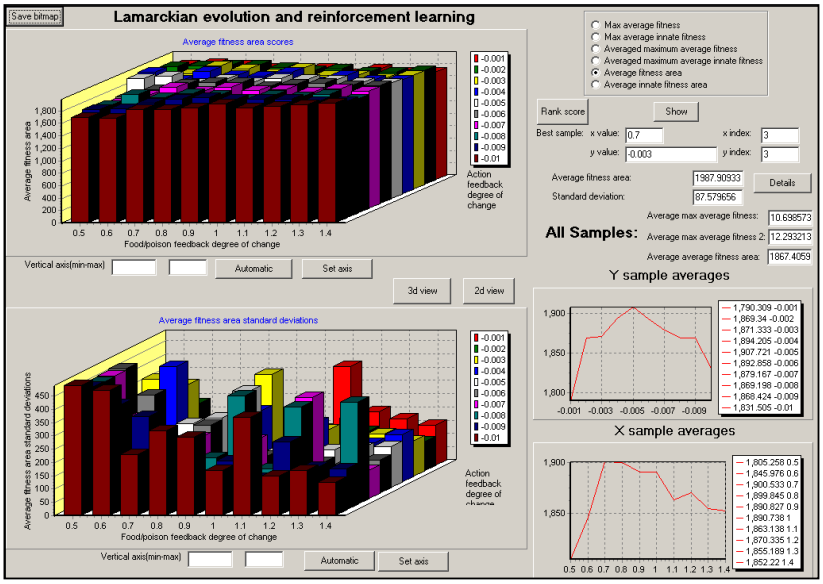


Figure L-5 3d perspective of average fitness area scores produced allowing food, poison and action feedback to be adjusted by degrees of change in the range 0.5 to 1.4 for food/poison and 0.001 to 0.01 for feedback per action.

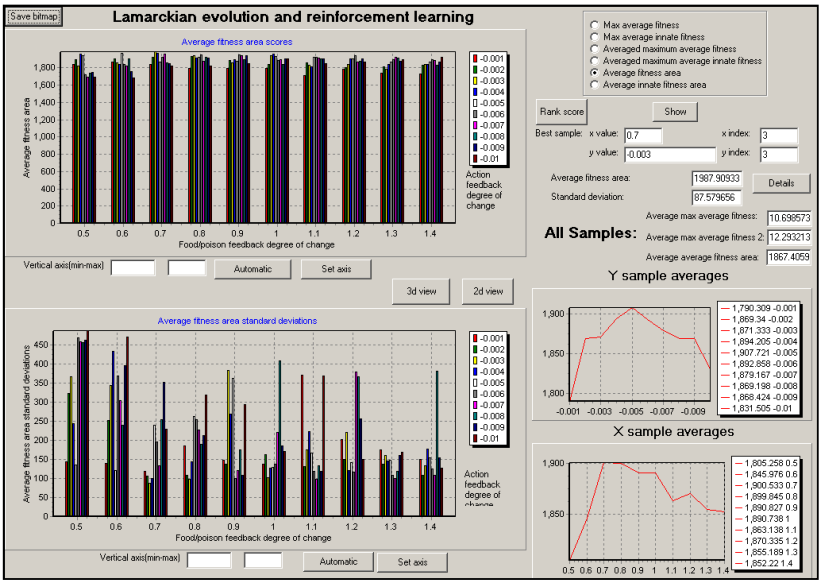


Figure L-6 2d perspective of average fitness area scores produced allowing food, poison and action feedback to be adjusted by degrees of change in the range 0.5 to 1.4 for food/poison and 0.001 to 0.01 for feedback per action.

Appendix M

Evaluation of Our Lamarckian Evolution and Reinforcement Learning Model No Mutations and Fixed Reinforcement Feedback Values in Stable to Highly Unstable Environments

The following data was produced using no mutations and fixed reinforcement feedback values of 3 for locating food, -3 for locating poison and –0.019 per action.

Maximum Average Fitness Scores

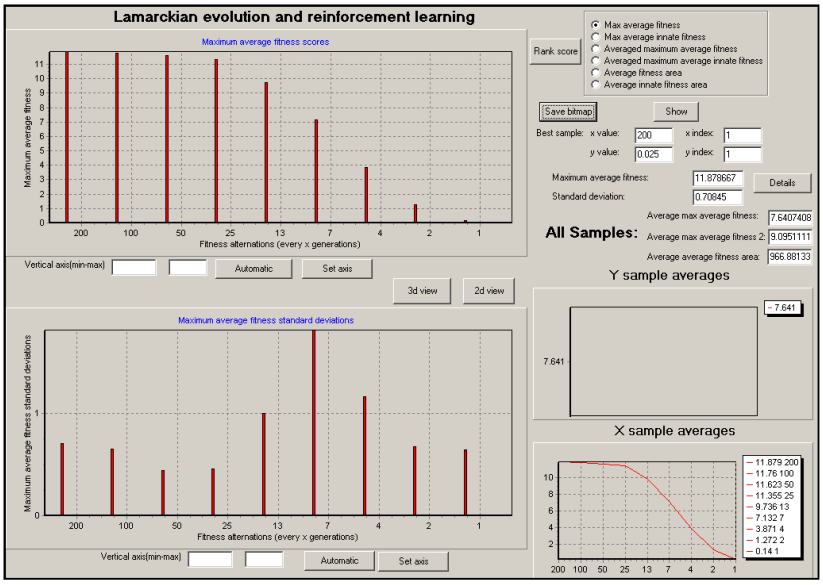


Figure M-1 Maximum average fitness scores produced by our Lamarckian evolution and reinforcement learning model in stable to highly unstable environments utilizing no mutations and fixed reinforcement feedback values.

Averaged Maximum Average Fitness Scores

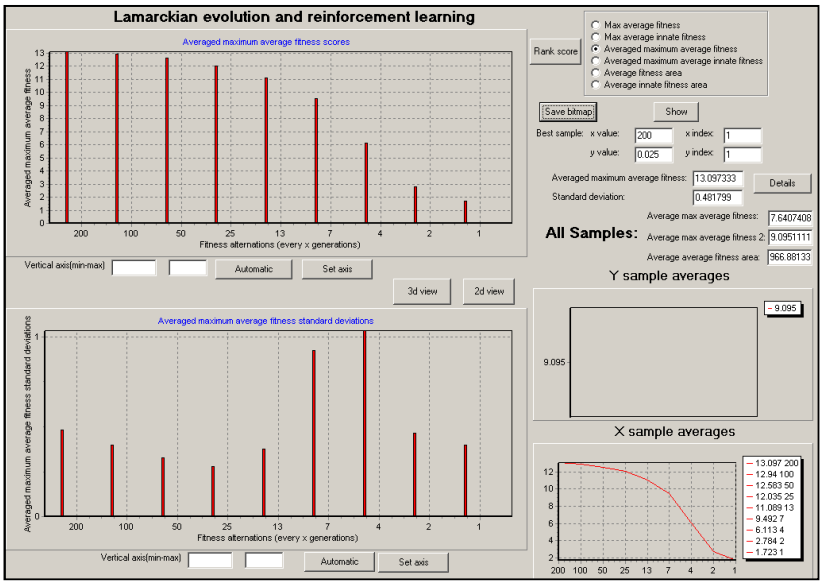


Figure M-2 Averaged maximum average fitness scores produced by our Lamarckian evolution and reinforcement learning model in stable to highly unstable environments utilizing no mutations and fixed reinforcement feedback values.

Average Fitness Area Scores

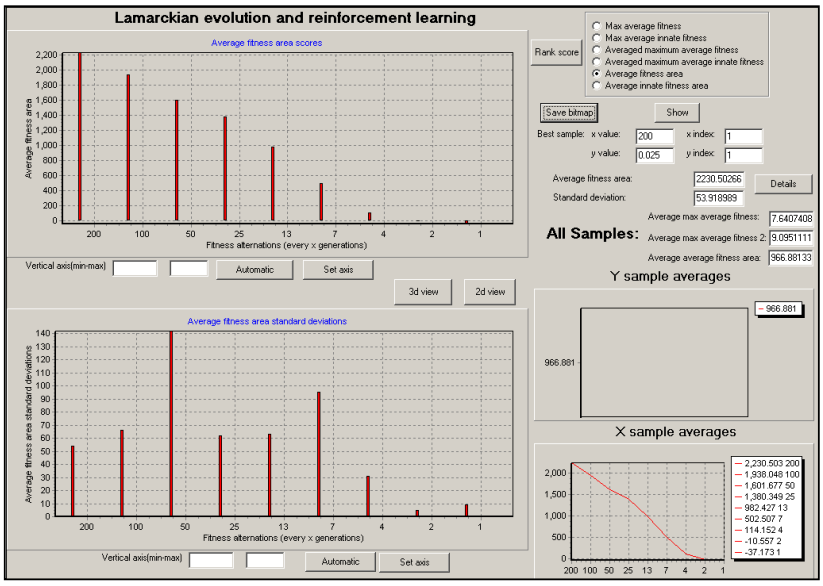


Figure M-3 Average fitness area scores produced by our Lamarckian evolution and reinforcement learning model in stable to highly unstable environments utilizing no mutations and fixed reinforcement feedback values.

Evaluation of Our Lamarckian Evolution and Reinforcement Learning Model With Fixed Mutation Rates and Fixed Reinforcement Feedback Values in Stable to Highly Unstable Environments

The following data was produced using a population mutation rate of 0.8 and a fixed individual mutation rate of 0.1 with fixed reinforcement feedback values of 3 for locating food, -3 for locating poison and –0.019 per action.

Maximum Average Fitness Scores

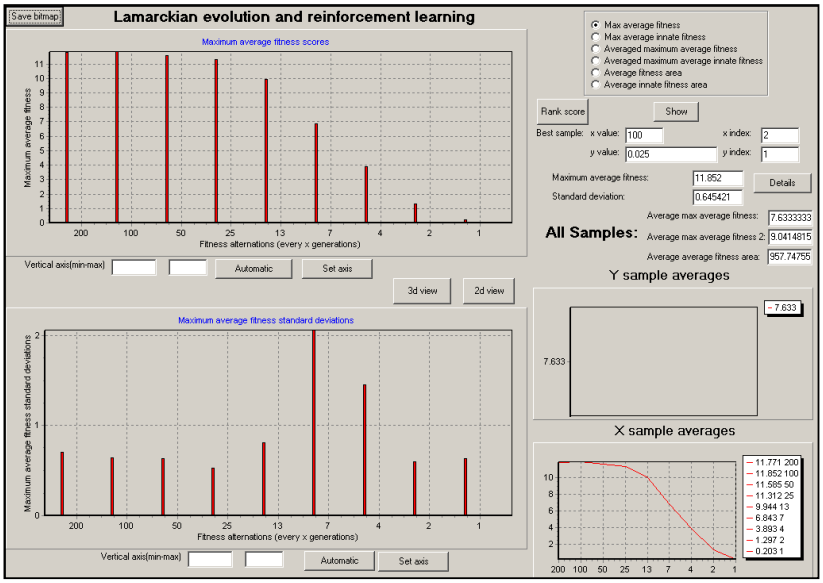


Figure M-4 Maximum average fitness scores produced by our Lamarckian evolution and reinforcement learning model in stable to highly unstable environments utilizing fixed mutation rates and fixed reinforcement feedback values.

Averaged Maximum Average Fitness Scores

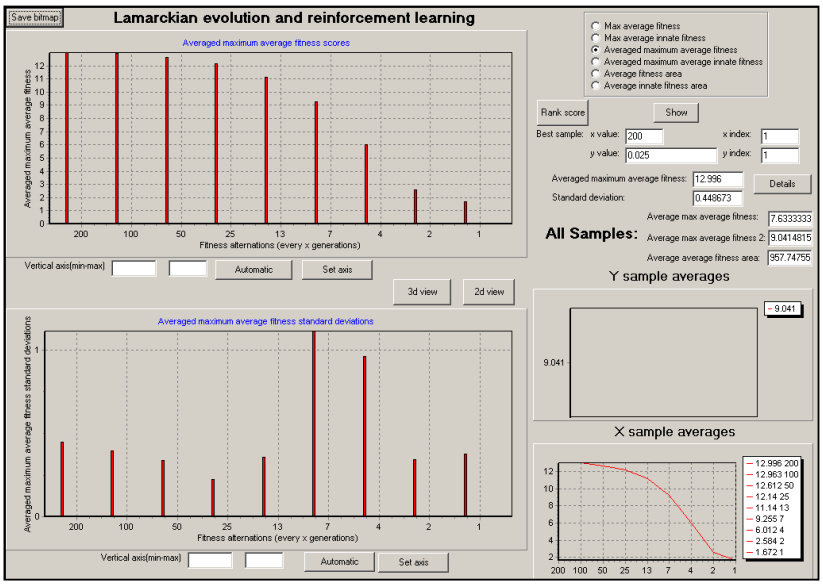


Figure M-5 Averaged maximum average fitness scores produced by our Lamarckian evolution and reinforcement learning model in stable to highly unstable environments utilizing fixed mutation rates and fixed reinforcement feedback values.

Average Fitness Area Scores

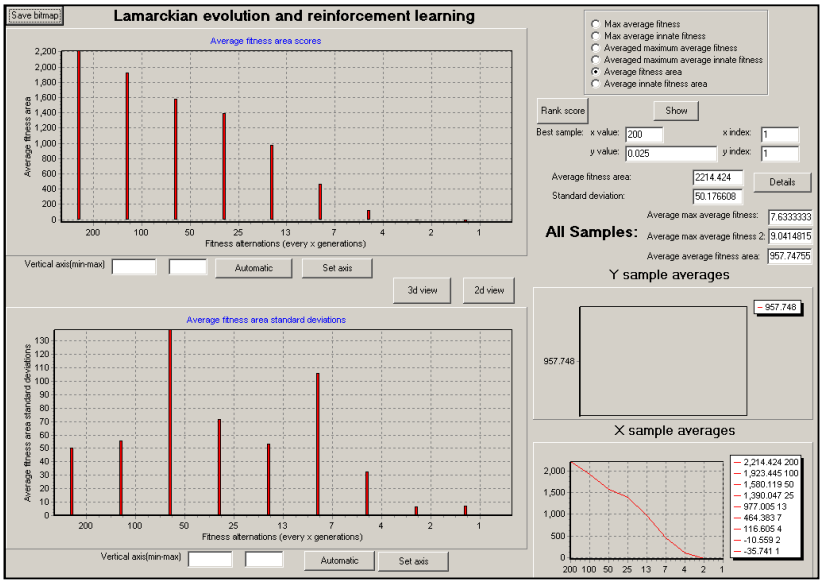


Figure M-6 Average fitness area scores produced by our Lamarckian evolution and reinforcement learning model in stable to highly unstable environments utilizing fixed mutation rates and fixed reinforcement feedback values.

Evaluation of Our Lamarckian Evolution and Reinforcement Learning Model Optimizing Individual Mutation Rates and Fixed Reinforcement Feedback Values in Stable to Highly Unstable Environments

The following data was produced utilizing a fixed population mutation rate of 0.8 and optimising individual mutation rates with fixed reinforcement feedback values of 3 for locating food, -3 for locating poison and –0.019 per action.

Maximum Average Fitness Scores

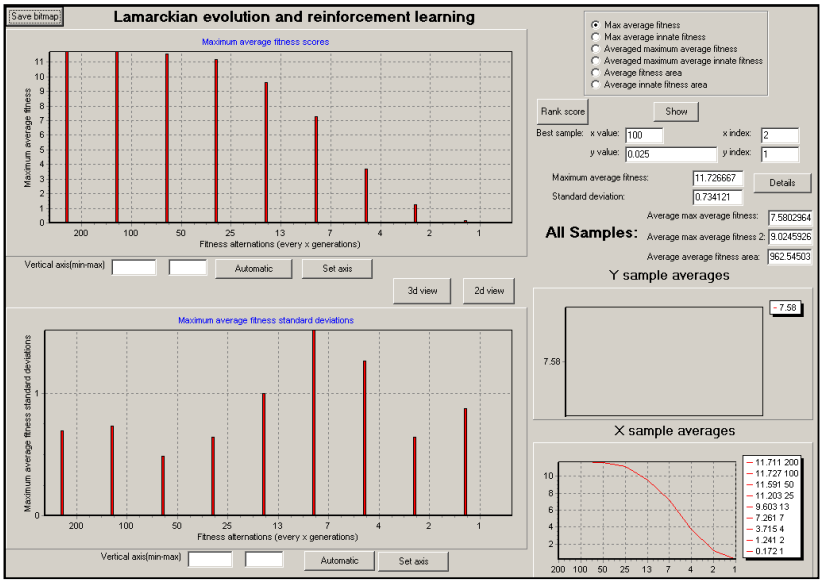


Figure M-7 Maximum average fitness scores produced by our Lamarckian evolution and reinforcement learning model in stable to highly unstable environments optimizing the individual mutation rate and utilizing fixed reinforcement feedback values.

Averaged Maximum Average Fitness Scores

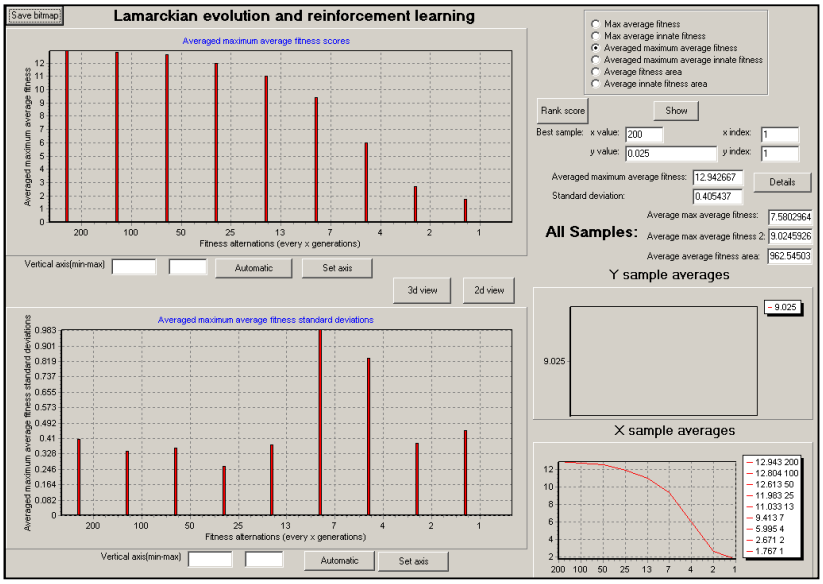


Figure M-8 Averaged maximum average fitness scores produced by our Lamarckian evolution and reinforcement learning model in stable to highly unstable environments optimizing the individual mutation rate and utilizing fixed reinforcement feedback values.

Average Fitness Area Scores

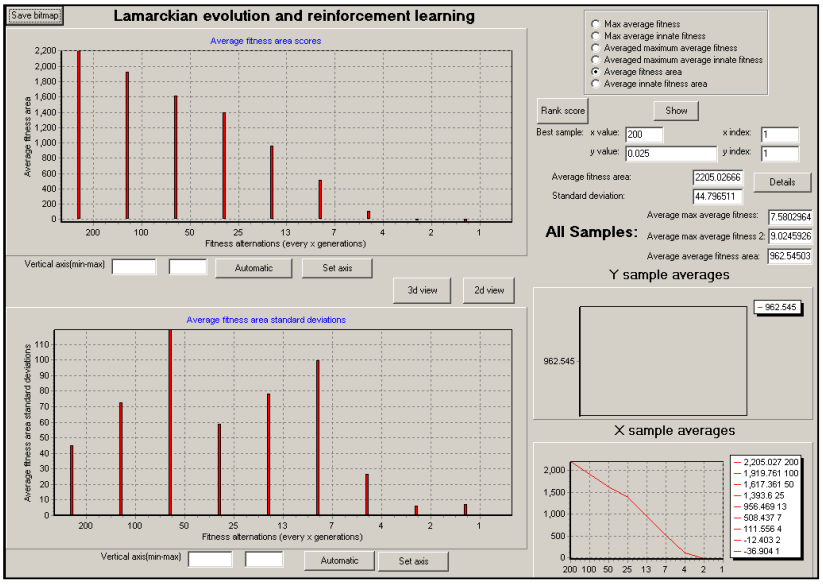


Figure M-9 Average fitness area scores produced by our Lamarckian evolution and reinforcement learning model in stable to highly unstable environments optimizing the individual mutation rate and utilizing fixed reinforcement feedback values.

Evaluation of Our Lamarckian Evolution and Reinforcement Learning Model No Mutations and Optimizing Reinforcement Feedback Values in Stable to Highly Unstable Environments

The following data was produced using no mutations and optimizing reinforcement feedback starting from initial values of 3 for locating food, -3 for locating poison and –0.019 per action.

Maximum Average Fitness Scores

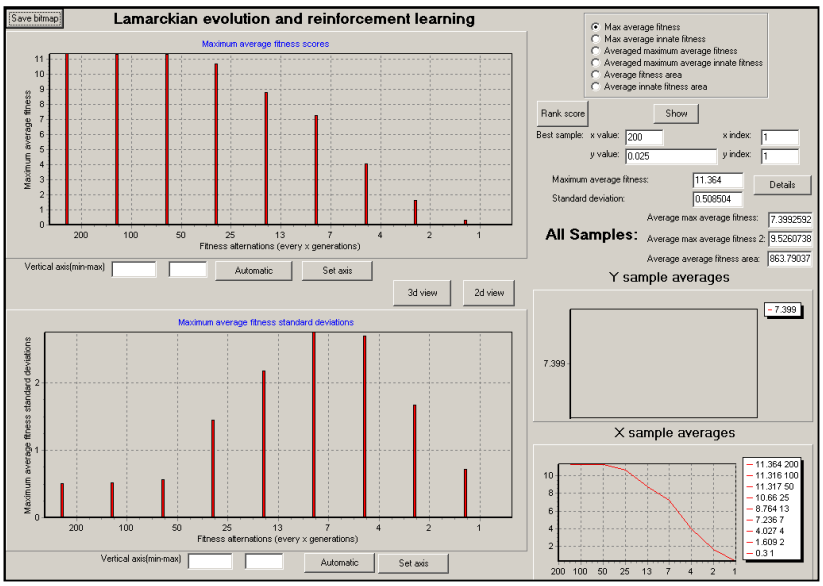


Figure M-10 Maximum average fitness scores produced by our Lamarckian evolution and reinforcement learning model in stable to highly unstable environments utilizing no mutations and optimizing reinforcement feedback values.

Averaged Maximum Average Fitness Scores

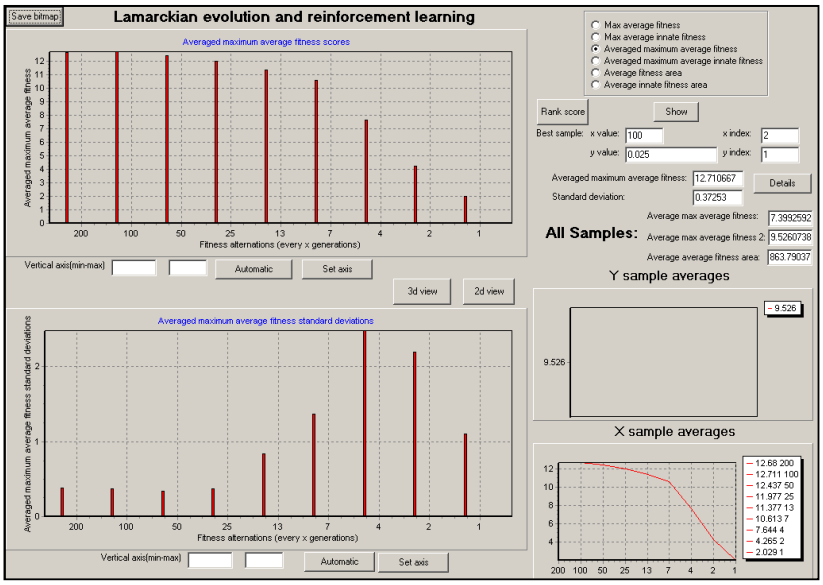


Figure M-11 Averaged maximum average fitness scores produced by our Lamarckian evolution and reinforcement learning model in stable to highly unstable environments utilizing no mutations and optimizing reinforcement feedback values.

Average Fitness Area Scores

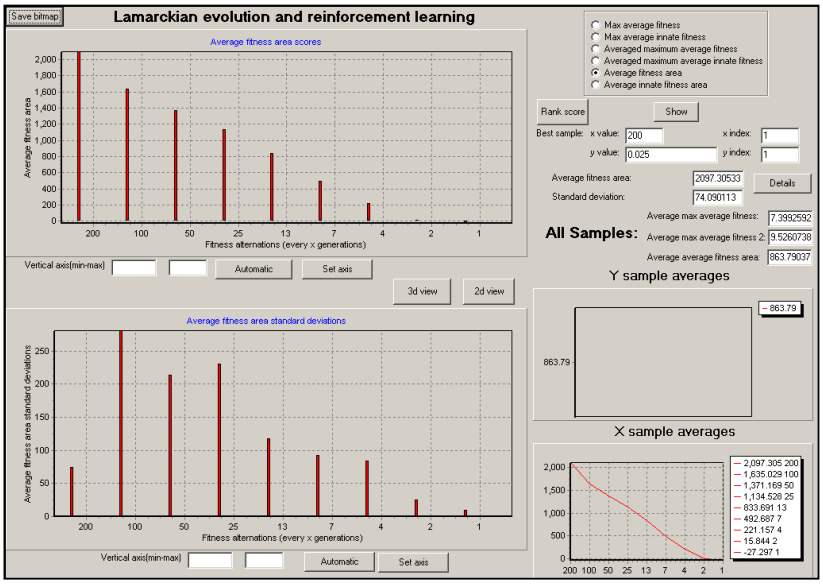


Figure M-12 Averaged fitness area scores produced by our Lamarckian evolution and reinforcement learning model in stable to highly unstable environments utilizing no mutations and optimizing reinforcement feedback values.

Evaluation of Our Lamarckian Evolution and Reinforcement Learning Model With Fixed Mutation Rates and Optimizing Reinforcement Feedback Values in Stable to Highly Unstable Environments

The following data was produced using a population mutation rate of 0.8 and a fixed individual mutation rate of 0.1 with reinforcement feedback being optimized from initial values of 3 for locating food, -3 for locating poison and –0.019 per action.

Maximum Average Fitness Scores

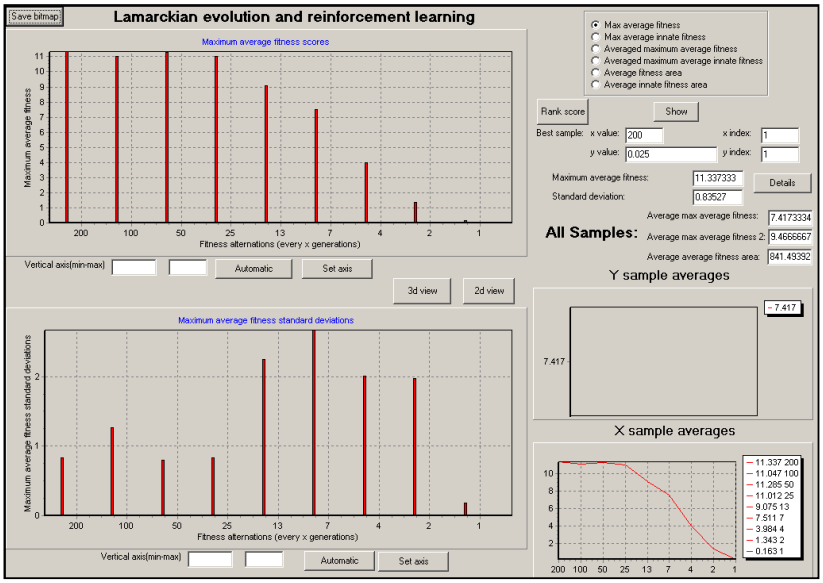


Figure M-13 Maximum average fitness scores produced by our Lamarckian evolution and reinforcement learning model in stable to highly unstable environments utilizing fixed mutation rates and optimizing reinforcement feedback values.

Averaged Maximum Average Fitness Scores

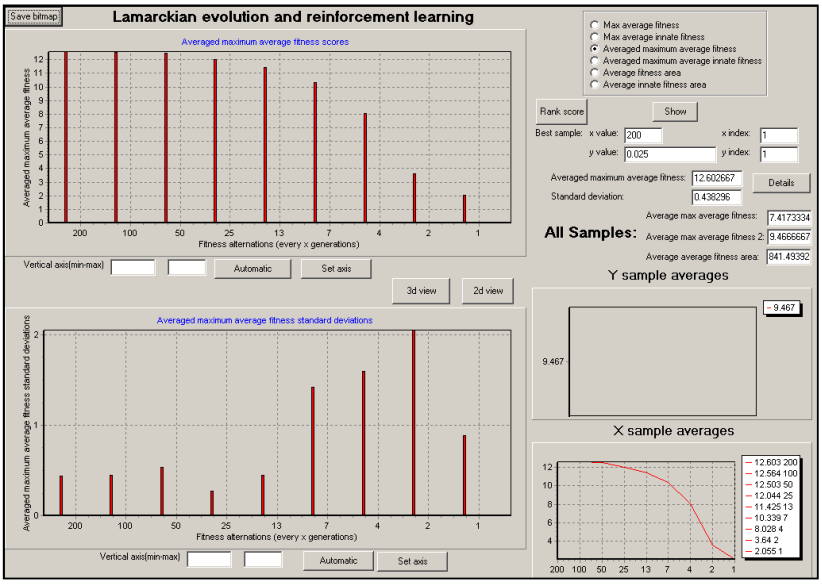


Figure M-14 Averaged maximum average fitness scores produced by our Lamarckian evolution and reinforcement learning model in stable to highly unstable environments utilizing fixed mutation rates and optimizing reinforcement feedback values.

Average Fitness Area Scores

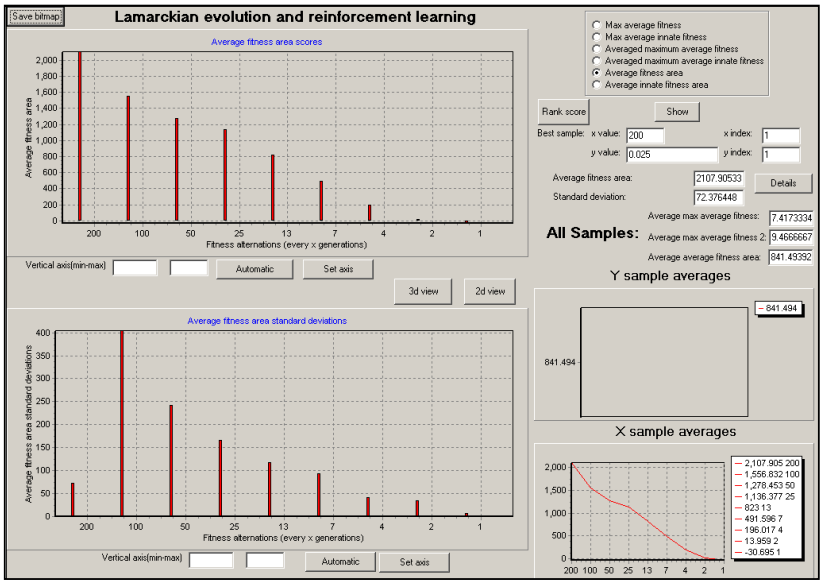


Figure M-15 Average fitness area scores produced by our Lamarckian evolution and reinforcement learning model in stable to highly unstable environments utilizing fixed mutation rates and optimizing reinforcement feedback values.

Evaluation of Our Lamarckian Evolution and Reinforcement Learning Model Optimizing Individual Mutation Rates and Reinforcement Feedback Values in Stable to Highly Unstable Environments

The following data was produced utilizing a fixed population mutation rate of 0.8 and optimising individual mutation rates with reinforcement feedback being optimized from initial starting values of 3 for locating food, -3 for locating poison and –0.019 per action.

Maximum Average Fitness Scores

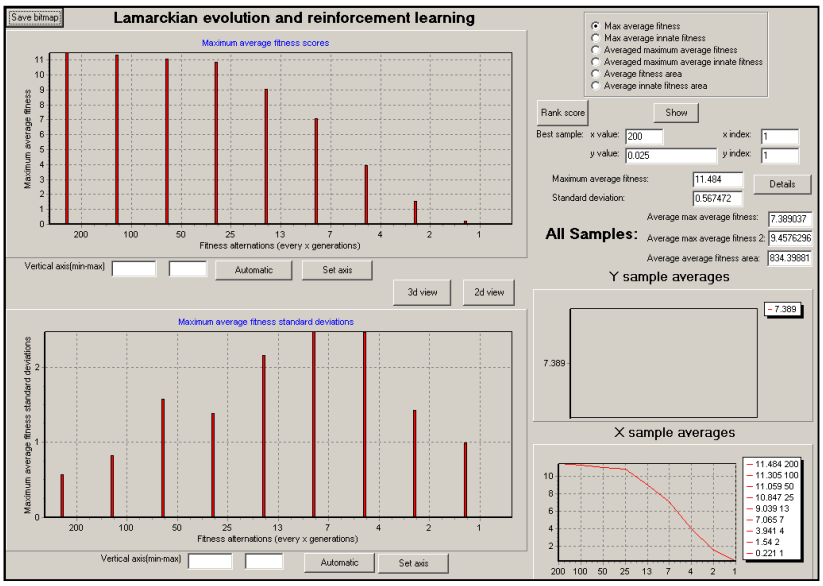


Figure M-16 Maximum average fitness scores produced by our Lamarckian evolution and reinforcement learning model in stable to highly unstable environments optimizing the individual mutation rate and reinforcement feedback values.

Averaged Maximum Average Fitness Scores

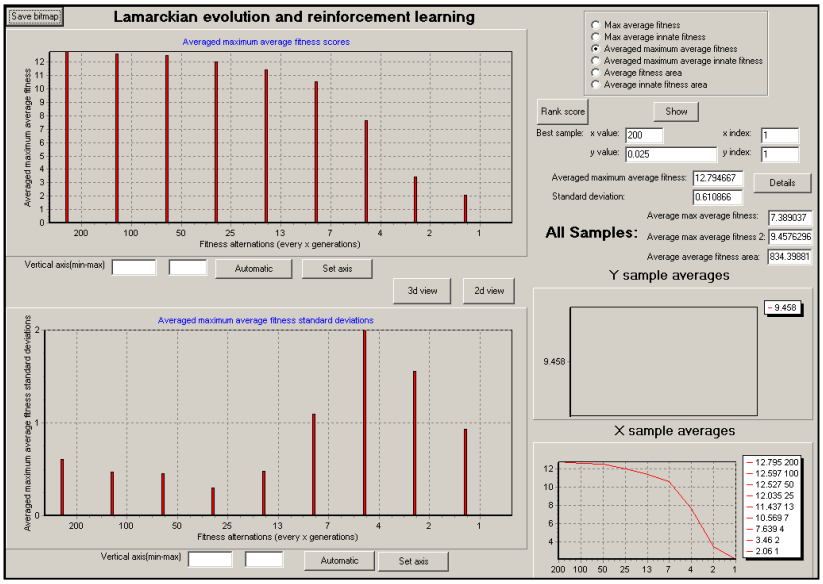


Figure M-17 Averaged maximum average fitness scores produced by our Lamarckian evolution and reinforcement learning model in stable to highly unstable environments optimizing the individual mutation rate and optimizing reinforcement feedback values.

Average Fitness Area Scores

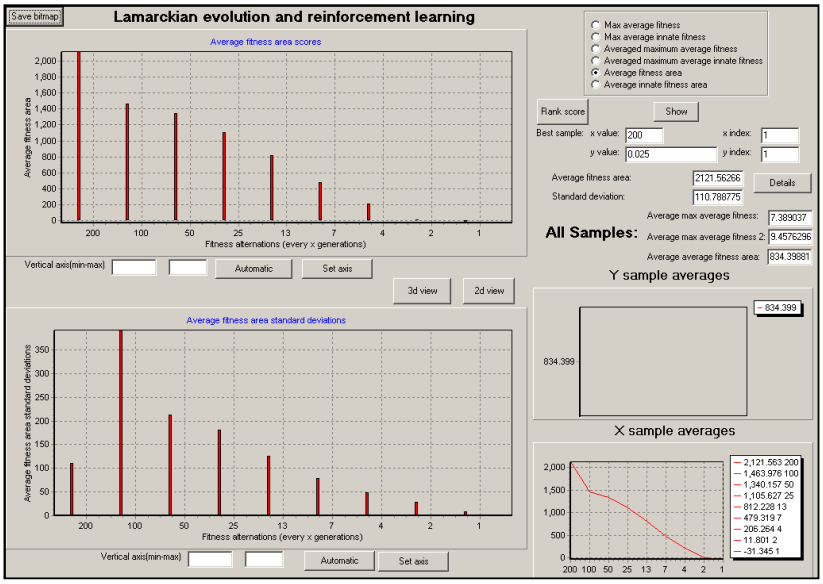


Figure M-18 Average fitness area scores produced by our Lamarckian evolution and reinforcement learning model in stable to highly unstable environments optimizing the individual mutation rate and utilizing fixed reinforcement feedback values.

Appendix N

Evaluation of Our Darwinian/Lamarckian Heterogeneous Population Model using Fixed Mutation Rates and Fixed Reinforcement Feedback Values in Stable to Highly Unstable Environments

The following data was produced using a fixed population mutation rate of 0.8 and an individual mutation rate of 0.1 with fixed reinforcement feedback values of 3 for locating food, -3 for locating poison and –0.019 per action.

Maximum Average Fitness Scores

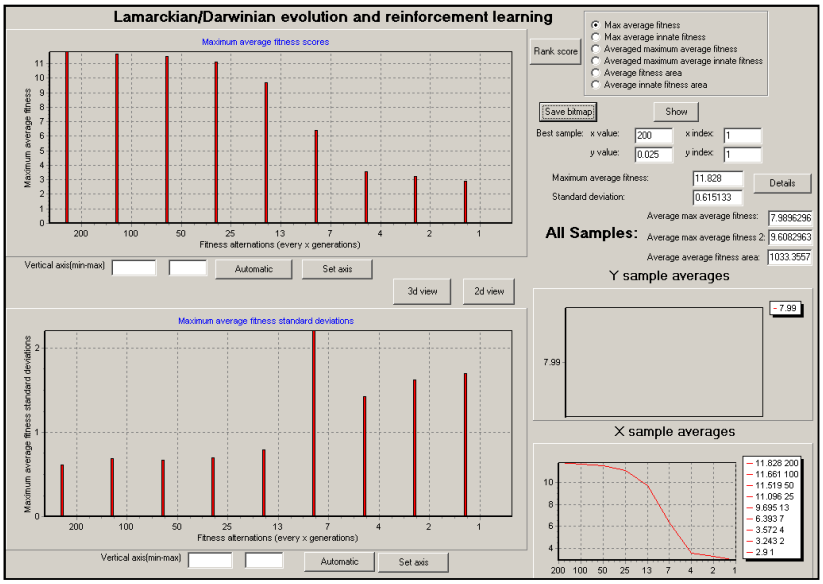


Figure N-1 Maximum average fitness scores produced by our Darwinian/Lamarckian heterogeneous population model in stable to highly unstable environments utilizing fixed mutation rates and fixed reinforcement feedback values.

Averaged Maximum Average Fitness Scores

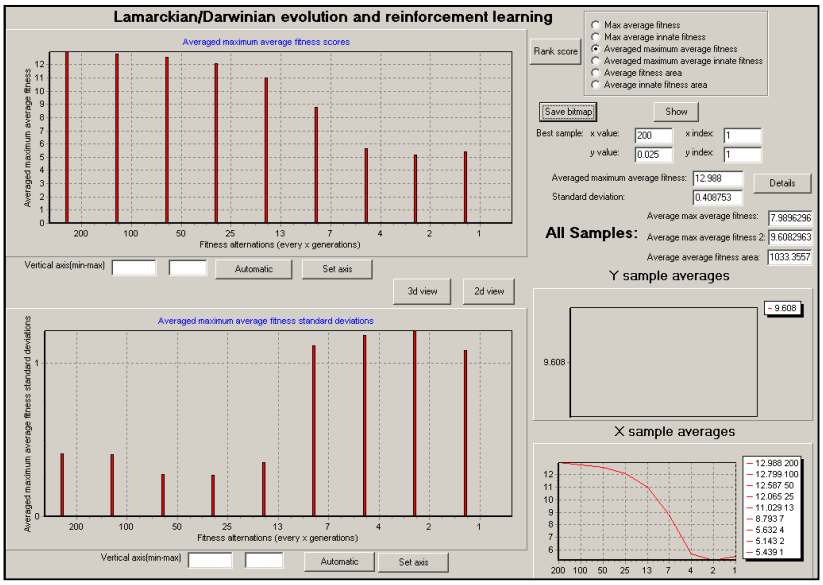


Figure N-2 Averaged maximum average fitness scores produced by our Darwinian/Lamarckian heterogeneous population model in stable to highly unstable environments utilizing fixed mutation rates and fixed reinforcement feedback values.

Average Fitness Area

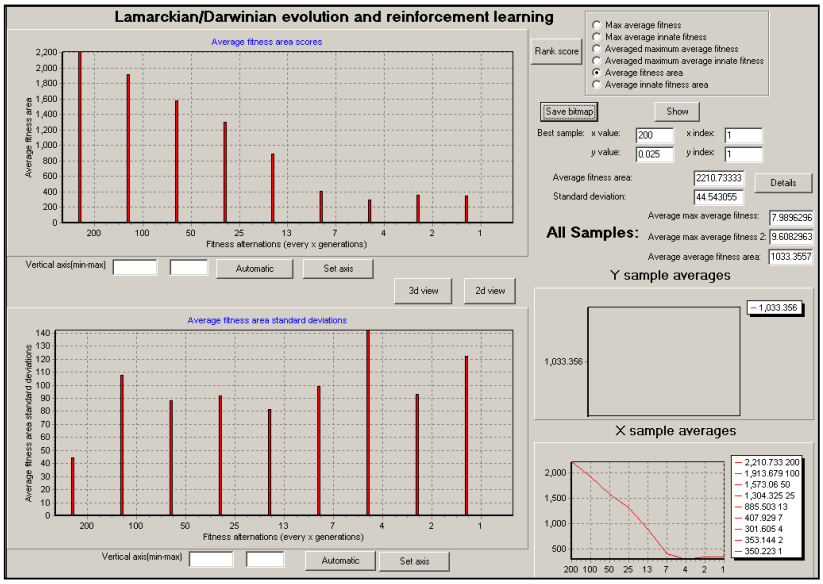


Figure N-3 Average fitness area scores produced by our Darwinian/Lamarckian heterogeneous population model in stable to highly unstable environments utilizing fixed mutation rates and fixed reinforcement feedback values.

Evaluation of Our Darwinian/Lamarckian Heterogeneous Population Model using Fixed Mutation Rates and Optimizing Reinforcement Feedback Values in Stable to Highly Unstable Environments

The following data was produced using a fixed population mutation rate of 0.8 and an individual mutation rate of 0.1 and optimising reinforcement feedback starting from initial values of 3 for locating food, -3 for locating poison and –0.019 per action.

Maximum Average Fitness Scores

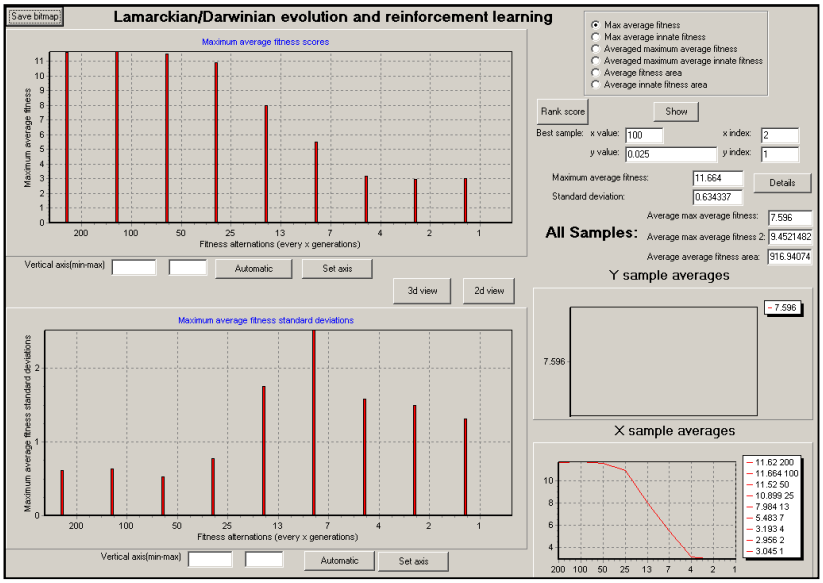


Figure N-4 Maximum average fitness scores produced by our Darwinian/Lamarckian heterogeneous population model in stable to highly unstable environments utilizing fixed mutation rates and optimizing reinforcement feedback values.

Averaged Maximum Average Fitness Scores

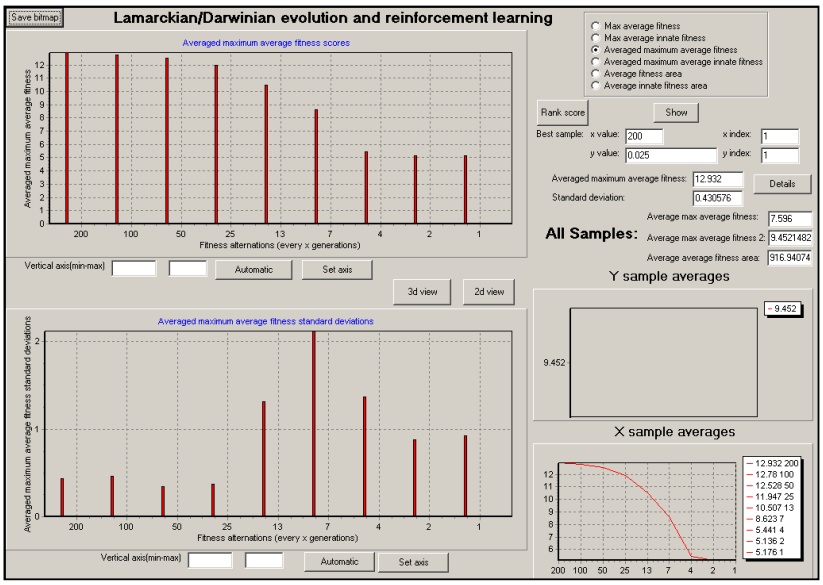


Figure N-5 Averaged maximum average fitness scores produced by our Darwinian/Lamarckian heterogeneous population model in stable to highly unstable environments utilizing fixed mutation rates and optimizing reinforcement feedback values.

Average Fitness Area Scores

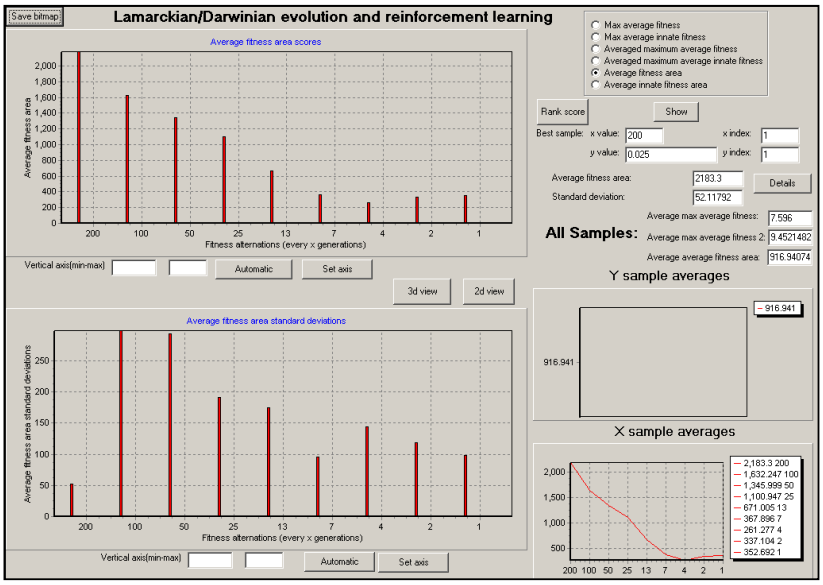


Figure N-6 Average fitness area scores produced by our Darwinian/Lamarckian heterogeneous population model in stable to highly unstable environments utilizing fixed mutation rates and optimizing reinforcement feedback values.

Evaluation of Our Darwinian/Lamarckian Heterogeneous Population Model Optimizing Mutation Rates and using Fixed Reinforcement Feedback Values in Stable to Highly Unstable Environments

The following data was produced using a fixed population mutation rate of 0.8 and optimizing individual mutation rates with fixed reinforcement feedback values of 3 for locating food, -3 for locating poison and –0.019 per action.

Maximum Average Fitness Scores

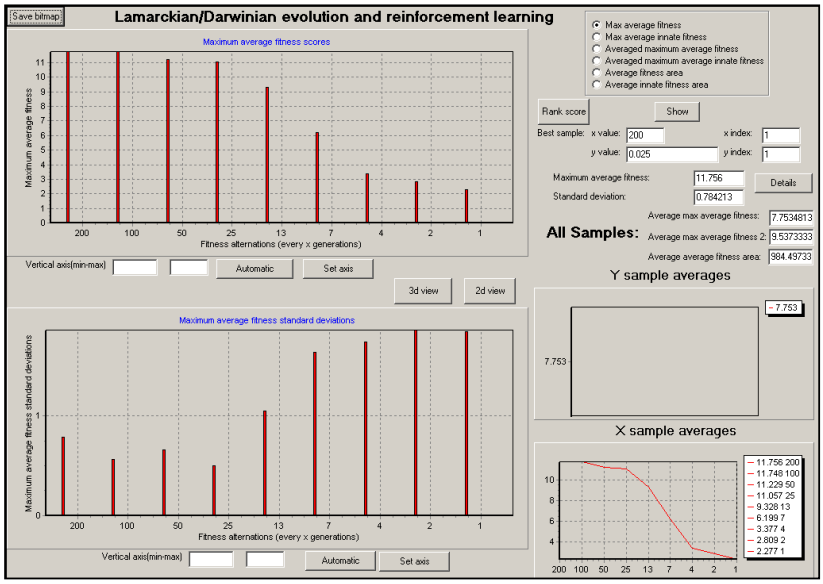


Figure N-7 Maximum average fitness scores produced by our Darwinian/Lamarckian heterogeneous population model in stable to highly unstable environments optimizing individual mutation rates with fixed reinforcement feedback values.

Averaged Maximum Average Fitness Scores

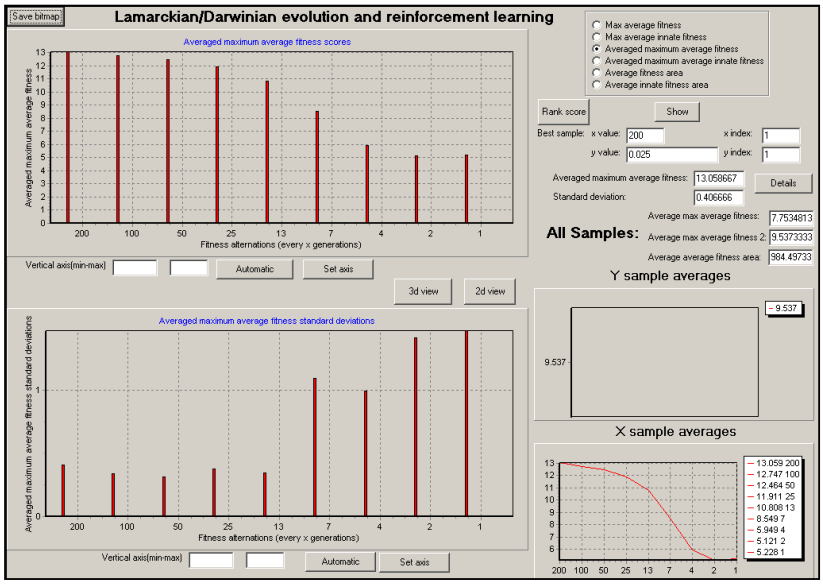


Figure N-8 Averaged maximum average fitness scores produced by our Darwinian/Lamarckian heterogeneous population model in stable to highly unstable environments optimizing individual mutation rates with fixed reinforcement feedback values.

Average Fitness Area Scores

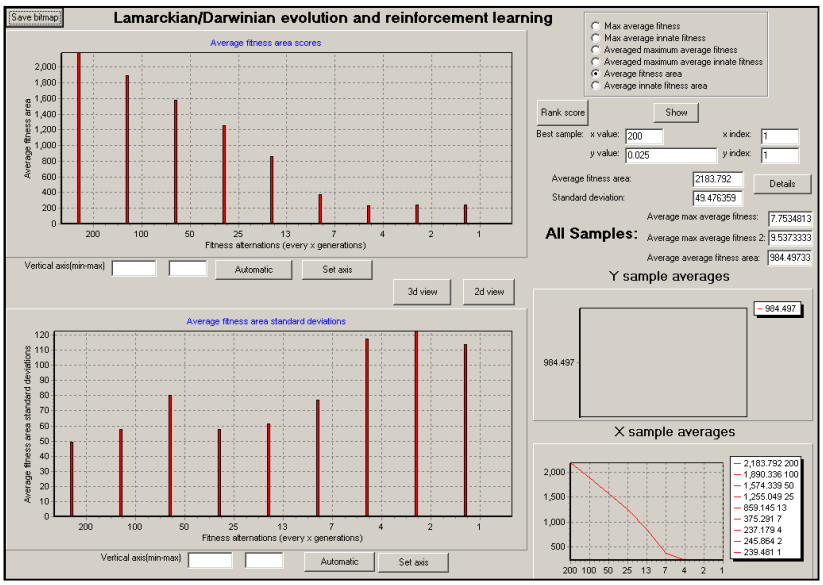


Figure N-9 Average fitness area scores produced by our Darwinian/Lamarckian heterogeneous population model in stable to highly unstable environments optimizing individual mutation rates with fixed reinforcement feedback values.

Evaluation of Our Darwinian/Lamarckian Heterogeneous Population Model Optimizing Mutation Rates and Reinforcement Feedback Values in Stable to Highly Unstable Environments

The following data was produced using a fixed population mutation rate of 0.8 and optimizing individual mutation rates and reinforcement feedback starting from initial values of 3 for locating food, -3 for locating poison and –0.019 per action.

Maximum Average Fitness Scores

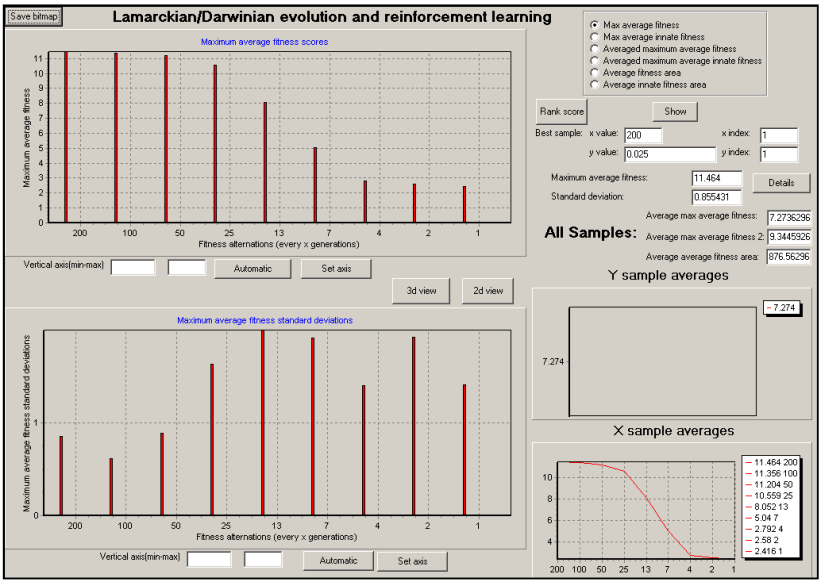


Figure N-10 Maximum average fitness scores produced by our Darwinian/Lamarckian heterogeneous population model in stable to highly unstable environments optimizing individual mutation rates and reinforcement feedback values.

Averaged Maximum Average Fitness Scores

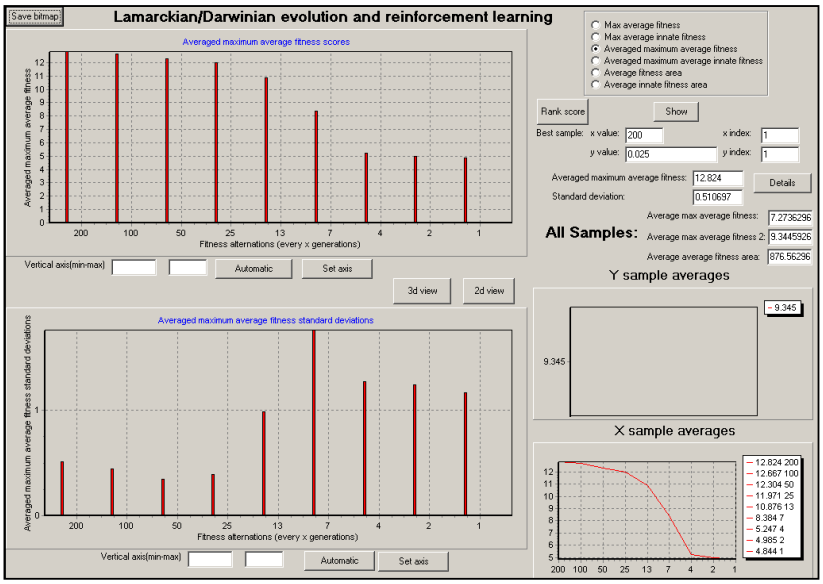


Figure N-11 Averaged maximum average fitness scores produced by our Darwinian/Lamarckian heterogeneous population model in stable to highly unstable environments optimizing individual mutation rates and reinforcement feedback values.

Average Fitness Area Scores

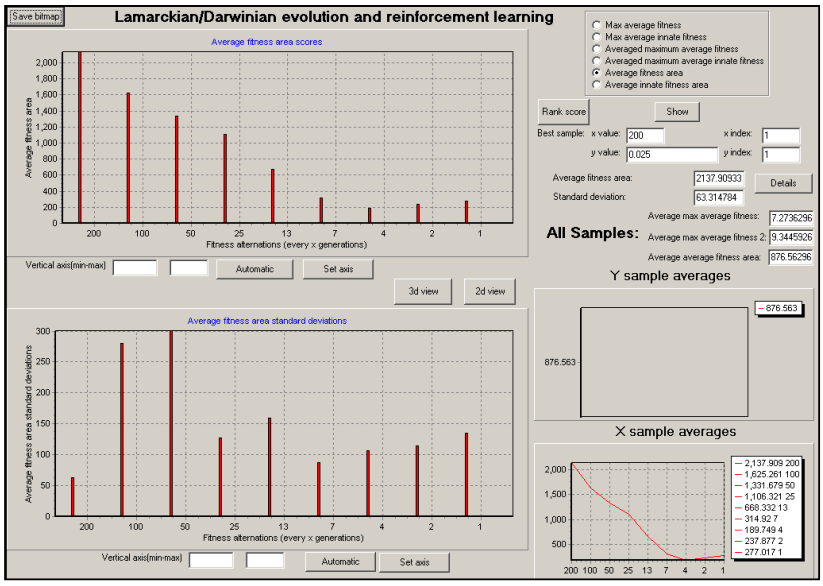


Figure N-12 Average fitness area scores produced by our Darwinian/Lamarckian heterogeneous population model in stable to highly unstable environments optimizing individual mutation rates and reinforcement feedback values.

Appendix O

Evaluation of Our Darwinian/Lamarckian Connection Model Optimizing Mutation Rates and Fixed Reinforcement Feedback Values in Stable to Highly Unstable Environments

The following data was produced using a fixed population mutation rate of 0.8 and optimizing the individual mutation rate with fixed reinforcement feedback values of 3 for locating food, -3 for locating poison and –0.019 per action.

Maximum Average Fitness Scores

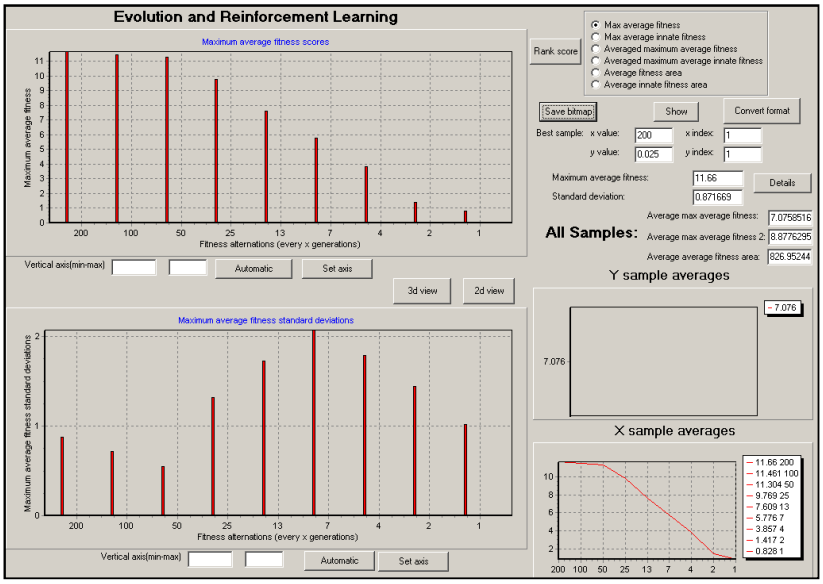


Figure O-1 Maximum average fitness scores produced by our Darwinian/Lamarckian connection model in stable to highly unstable environments optimizing mutation rates and using fixed reinforcement feedback values.

Averaged Maximum Average Fitness Scores

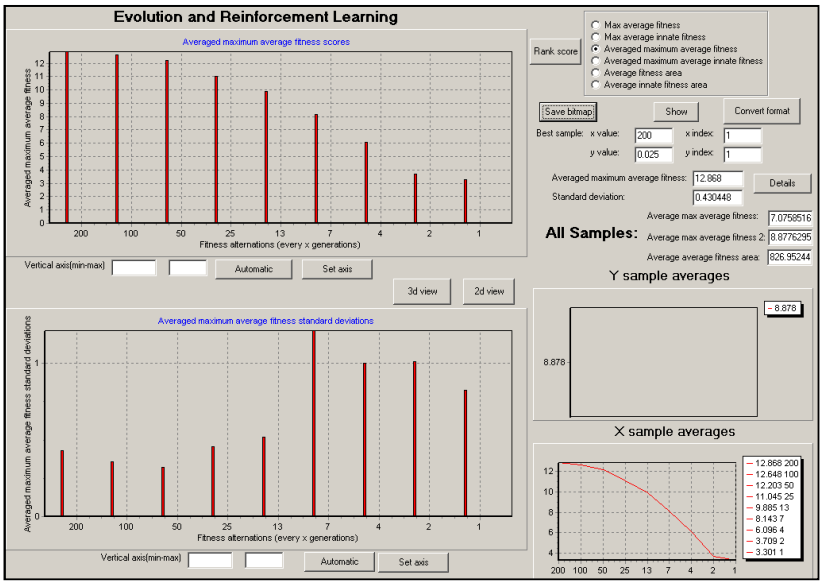


Figure O-2 Averaged maximum average fitness scores produced by our Darwinian/Lamarckian connection model in stable to highly unstable environments optimizing mutation rates and using fixed reinforcement feedback values.

Average Fitness Area Scores

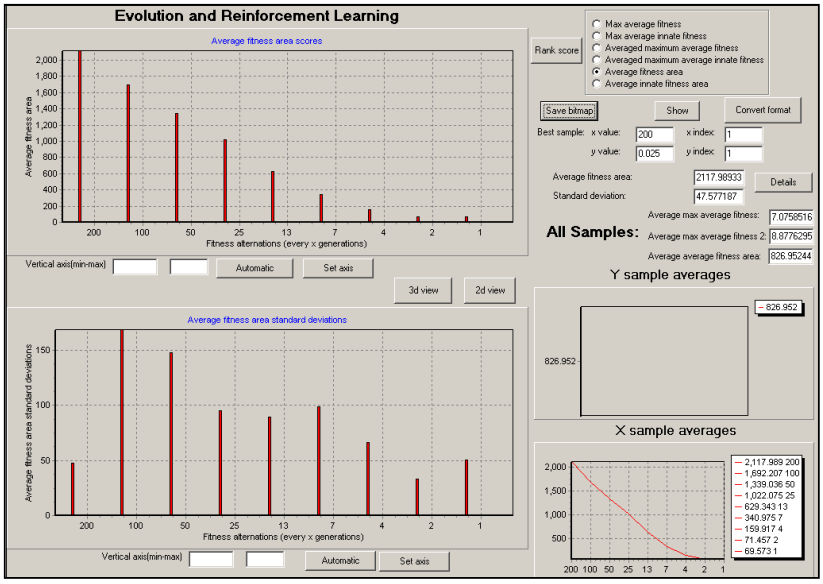


Figure O-3 Average fitness area scores produced by our Darwinian/Lamarckian connection model in stable to highly unstable environments optimizing mutation rates and using fixed reinforcement feedback values.

Evaluation of Our Darwinian/Lamarckian Connection Model using Fixed Mutation Rates and Fixed Reinforcement Feedback Values in Stable to Highly Unstable Environments

The following data was produced using a fixed population mutation rate of 0.8 and a fixed individual mutation rate of 0.1 with fixed reinforcement feedback values of 3 for locating food, -3 for locating poison and –0.019 per action.

Maximum Average Fitness Scores

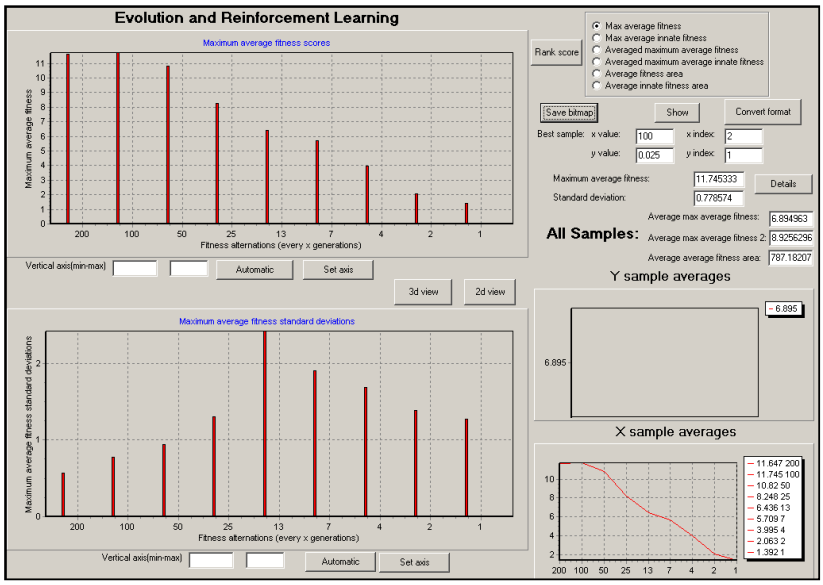


Figure O-4 Maximum average fitness scores produced by our Darwinian/Lamarckian connection model in stable to highly unstable environments utilizing fixed mutation rates and fixed reinforcement feedback values.

Averaged Maximum Average Fitness Scores

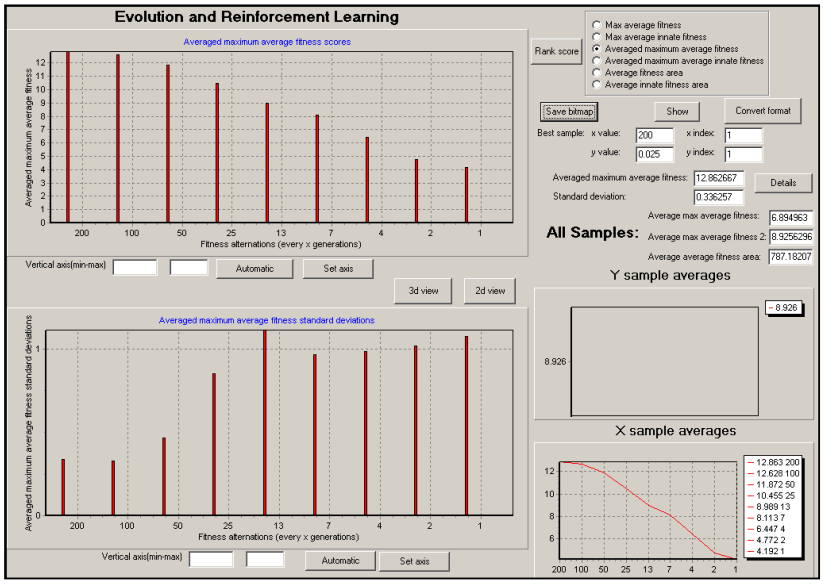


Figure O-5 Averaged maximum average fitness scores produced by our Darwinian/Lamarckian connection model in stable to highly unstable environments utilizing fixed mutation rates and fixed reinforcement feedback values.

Average Fitness Area Scores

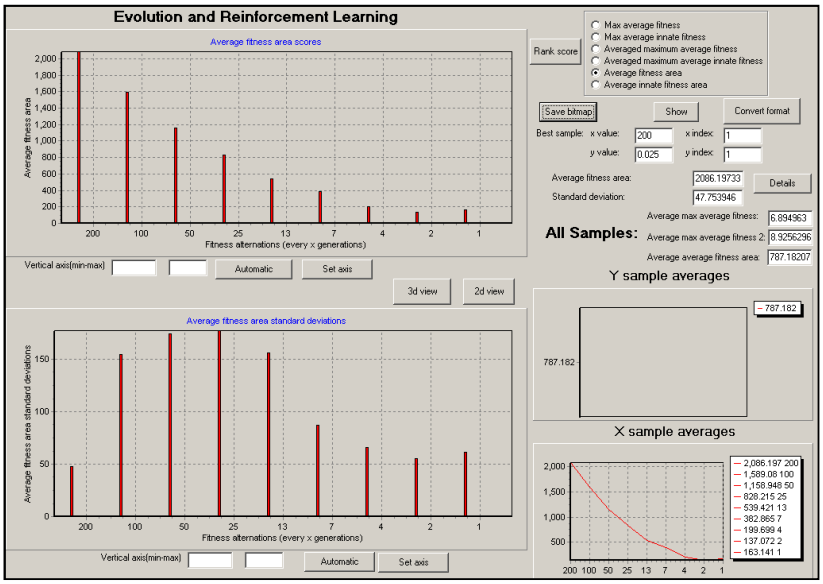


Figure O-6 Average fitness area scores produced by our Darwinian/Lamarckian connection model in stable to highly unstable environments utilizing fixed mutation rates and fixed reinforcement feedback values.

Evaluation of Our Darwinian/Lamarckian Connection Model using Fixed Mutation Rates and Optimizing Reinforcement Feedback Values in Stable to Highly Unstable Environments

The following data was produced using a fixed population mutation rate of 0.8 and a fixed individual mutation rate of 0.1 and optimizing reinforcement feedback starting from initial values of 3 for locating food, -3 for locating poison and –0.019 per action.

Maximum Average Fitness Scores

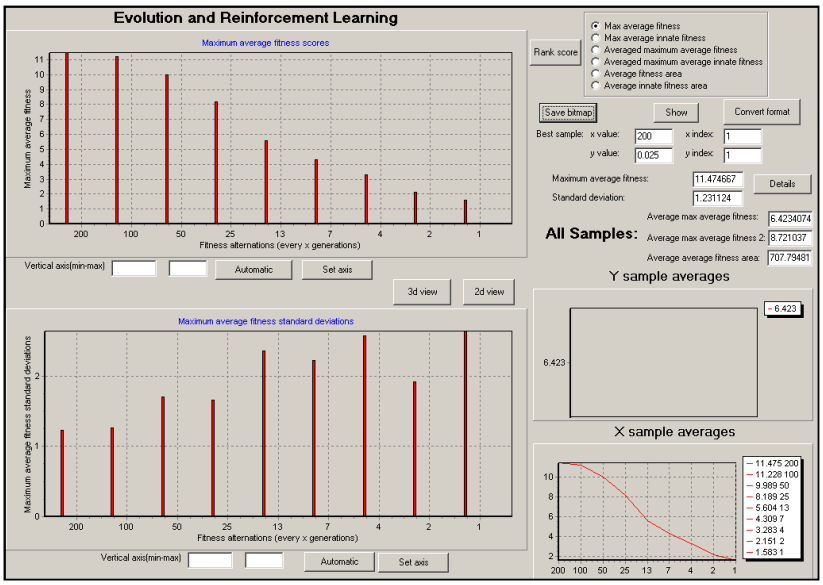


Figure O-7 Maximum average fitness scores produced by our Darwinian/Lamarckian connection model in stable to highly unstable environments utilizing fixed mutation rates and optimizing reinforcement feedback values.

Averaged Maximum Average Fitness Scores

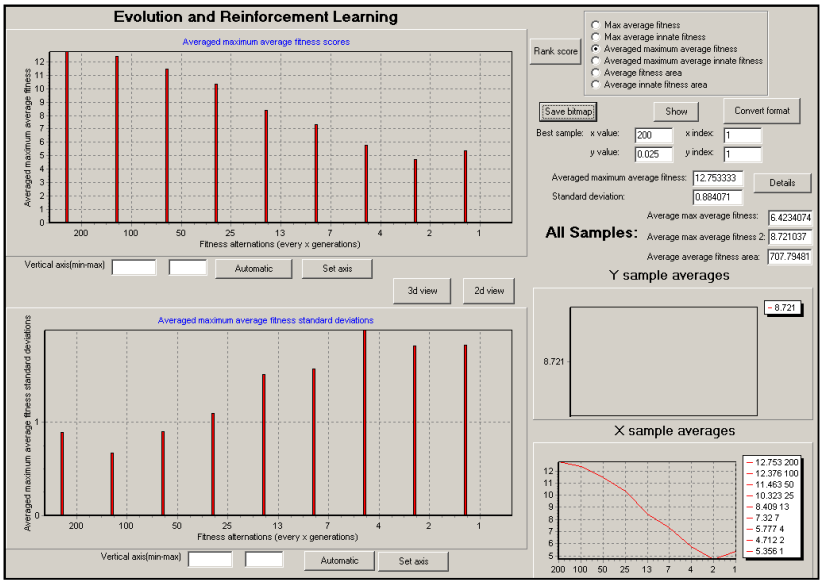


Figure O-8 Averaged maximum average fitness scores produced by our Darwinian/Lamarckian connection model in stable to highly unstable environments utilizing fixed mutation rates and optimizing reinforcement feedback values.

Average Fitness Area Scores

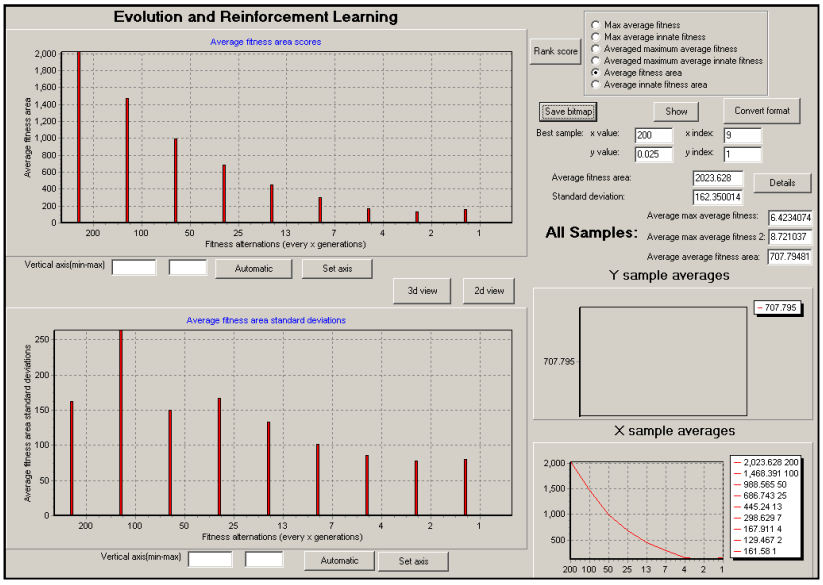


Figure O-9 Average fitness area scores produced by our Darwinian/Lamarckian connection model in stable to highly unstable environments utilizing fixed mutation rates and optimizing reinforcement feedback values.

Evaluation of Our Darwinian/Lamarckian Connection Model Optimizing Mutation Rates and Optimizing Reinforcement Feedback Values in Stable to Highly Unstable Environments

The following data was produced using a fixed population mutation rate of 0.8 and optimizing individual mutation rates and reinforcement feedback starting from initial values of 3 for locating food, -3 for locating poison and –0.019 per action.

Maximum Average Fitness Scores

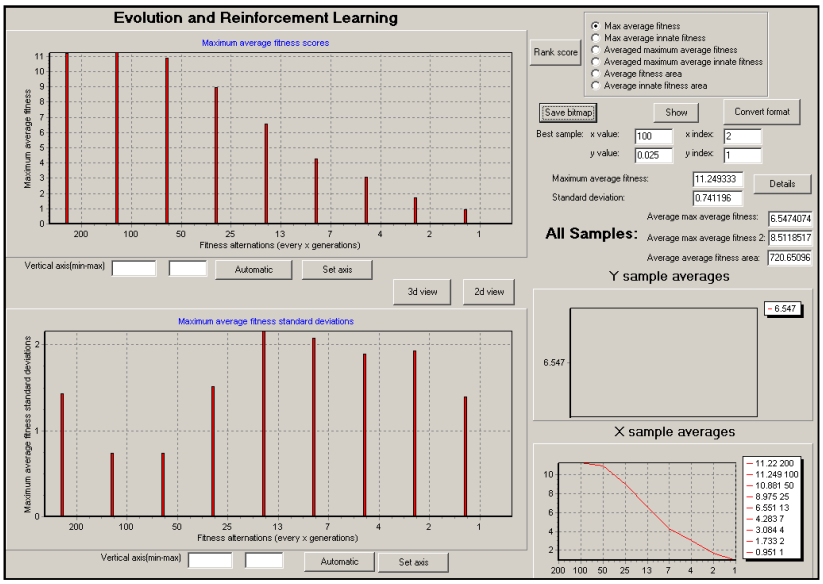


Figure O-10 Maximum average fitness scores produced by our Darwinian/Lamarckian connection model in stable to highly unstable environments optimizing mutation rates and reinforcement feedback values.

Averaged Maximum Average Fitness Scores

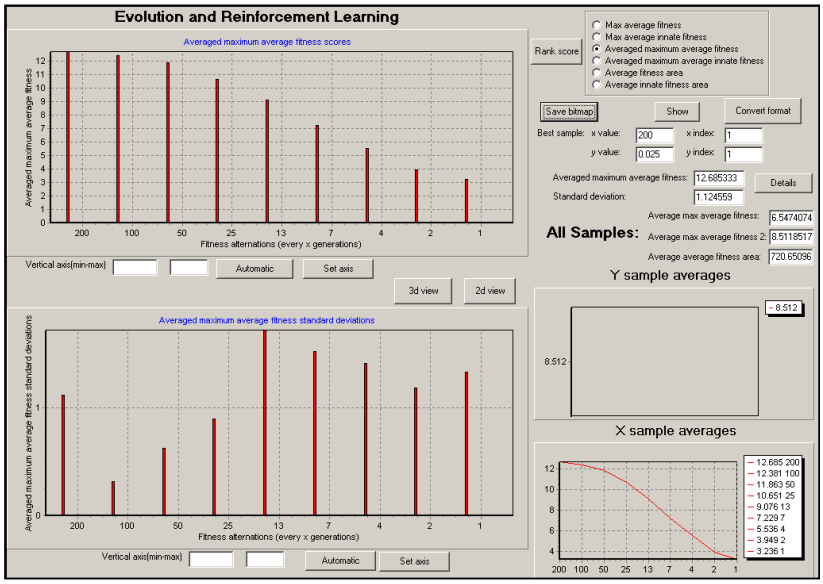


Figure O-11 Averaged maximum average fitness scores produced by our Darwinian/Lamarckian connection model in stable to highly unstable environments optimizing mutation rates and reinforcement feedback values.

Average Fitness Area Scores

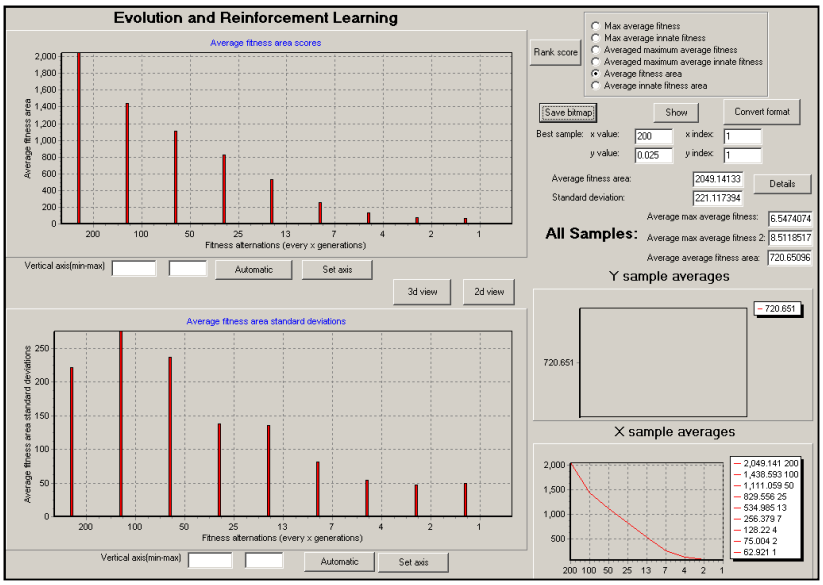


Figure O-12 Average fitness area scores produced by our Darwinian/Lamarckian connection model in stable to highly unstable environments optimizing mutation rates and reinforcement feedback values.

Appendix P

Evaluation of Our Darwinian/Lamarckian Global Degree of Neural Network Inheritance Model Utilizing Fixed Mutation Rates and Fixed Reinforcement Feedback Values in Stable to Highly Unstable Environments

The following data was produced using a fixed population mutation rate of 0.8 and fixed individual mutation rate of 0.1 with fixed reinforcement feedback values of 3 for locating food, -3 for locating poison and –0.019 per action.

Maximum Average Fitness Scores

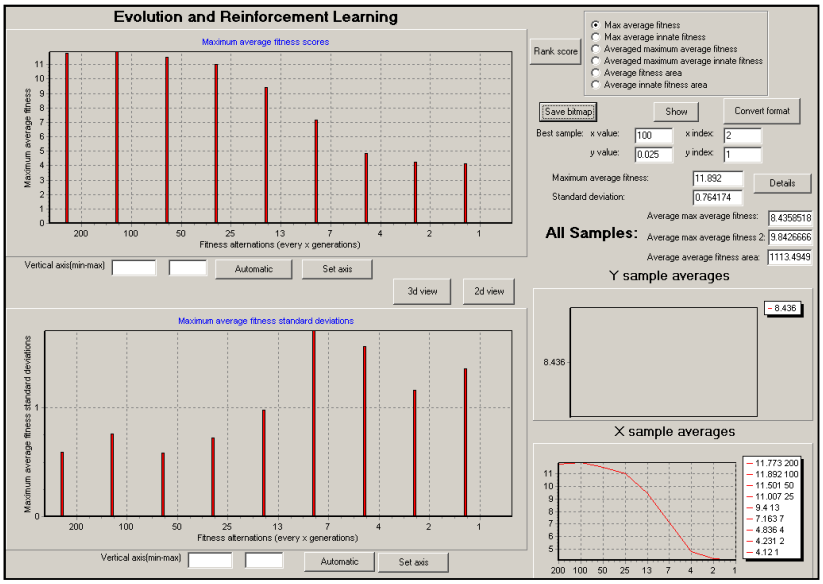


Figure P-1 Maximum average fitness scores produced by our Darwinian/Lamarckian global degree of neural network inheritance model in stable to highly unstable environments utilizing fixed mutation rates and fixed reinforcement feedback values.

Averaged Maximum Average Fitness Scores

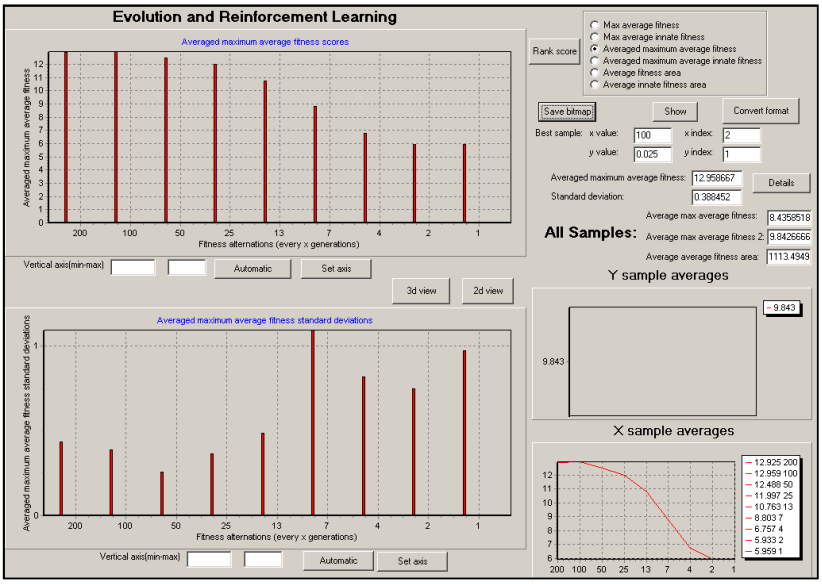


Figure P-2 Averaged maximum average fitness scores produced by our Darwinian/Lamarckian global degree of inheritance model in stable to highly unstable environments utilizing fixed mutation rates and fixed reinforcement feedback values.

Average Fitness Area Scores

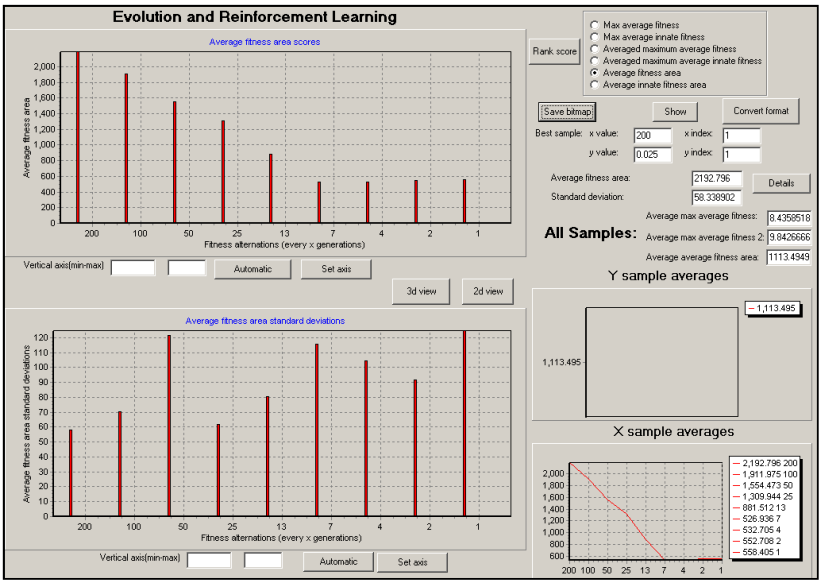


Figure P-3 Average fitness area scores produced by our Darwinian/Lamarckian global degree of inheritance model in stable to highly unstable environments utilizing fixed mutation rates and fixed reinforcement feedback values.

Evaluation of Our Darwinian/Lamarckian Global Degree of Neural Network Inheritance Model Utilizing Fixed Mutation Rates and Optimizing Reinforcement Feedback Values in Stable to Highly Unstable Environments

The following data was produced using a fixed population mutation rate of 0.8 and fixed individual mutation rate of 0.1 and optimizing reinforcement feedback starting from initial values of 3 for locating food, -3 for locating poison and –0.019 per action.

Maximum Average Fitness Scores

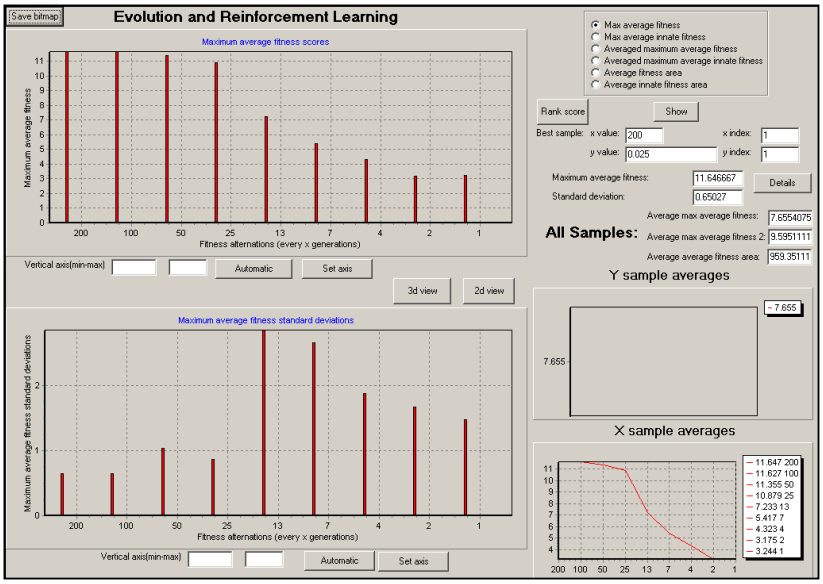


Figure P-4 Maximum average fitness scores produced by our Darwinian/Lamarckian global degree of neural network inheritance model in stable to highly unstable environments utilizing fixed mutation rates and optimizing reinforcement feedback values.

Averaged Maximum Average Fitness Scores

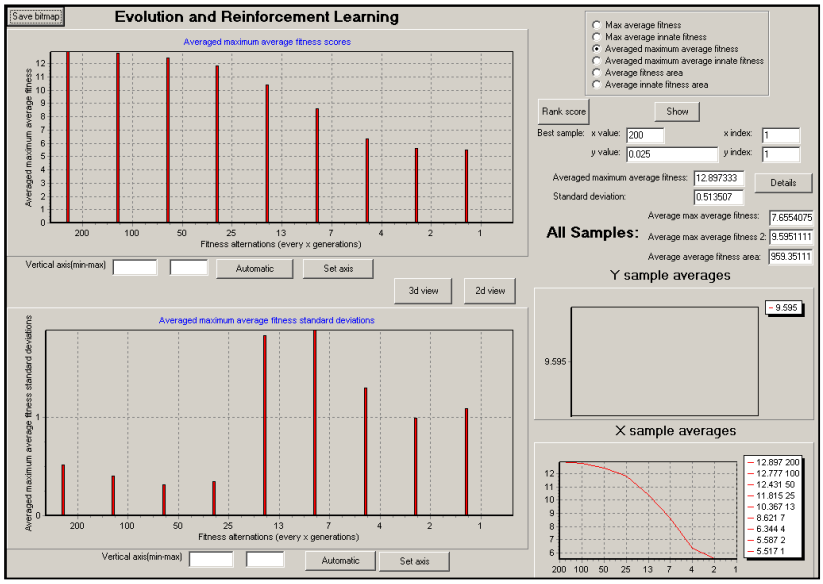


Figure P-5 Averaged maximum average fitness scores produced by our Darwinian/Lamarckian global degree of neural network inheritance model in stable to highly unstable environments utilizing fixed mutation rates and optimizing reinforcement feedback values.

Average Fitness Area Scores

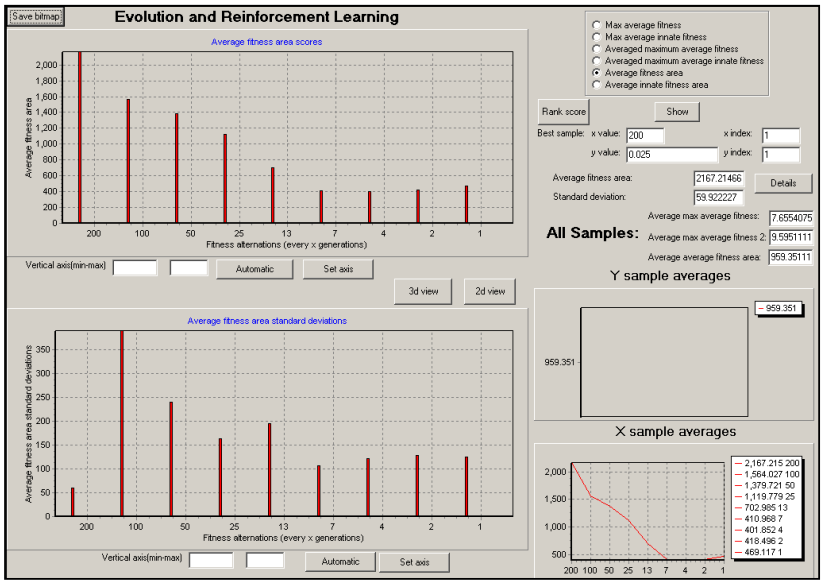


Figure P-6 Average fitness area scores produced by our Darwinian/Lamarckian global degree of neural network inheritance model in stable to highly unstable environments utilizing fixed mutation rates and optimizing reinforcement feedback values.

Evaluation of Our Darwinian/Lamarckian Global Degree of Neural Network Inheritance Model Optimizing Mutation Rates and using Fixed Reinforcement Feedback Values in Stable to Highly Unstable Environments

The following data was produced using a fixed population mutation rate of 0.8 and optimizing individual mutation rates with fixed reinforcement feedback values of 3 for locating food, -3 for locating poison and –0.019 per action.

Maximum Average Fitness Scores

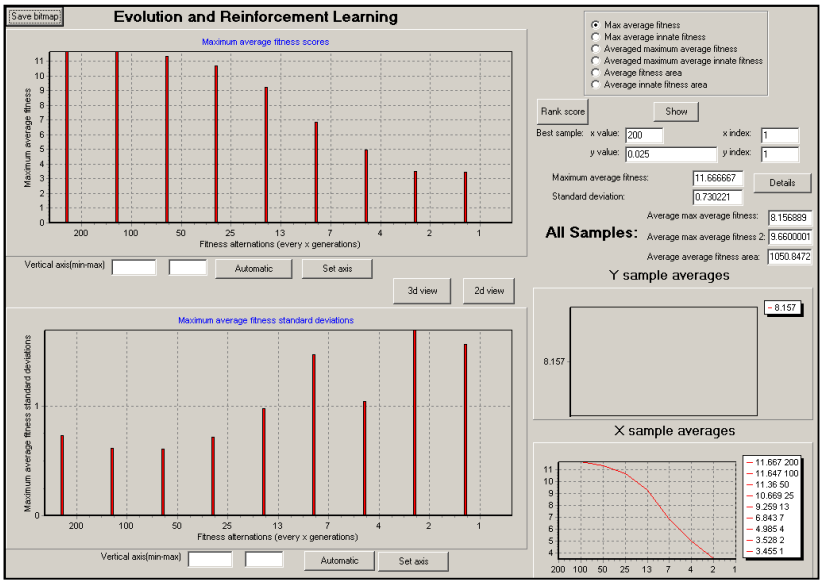


Figure P-7 Maximum average fitness scores produced by our Darwinian/Lamarckian global degree of neural network inheritance model in stable to highly unstable environments optimizing individual mutation rates with fixed reinforcement feedback values.

Averaged Maximum Average Fitness Scores

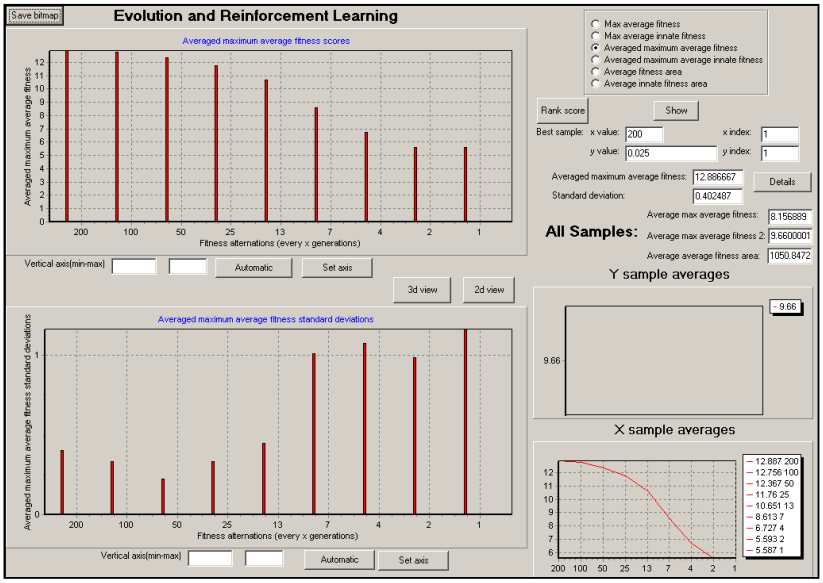


Figure P-8 Averaged maximum average fitness scores produced by our Darwinian/Lamarckian global degree of neural network inheritance model in stable to highly unstable environments optimizing individual mutation rates with fixed reinforcement feedback values.

Average Fitness Area Scores

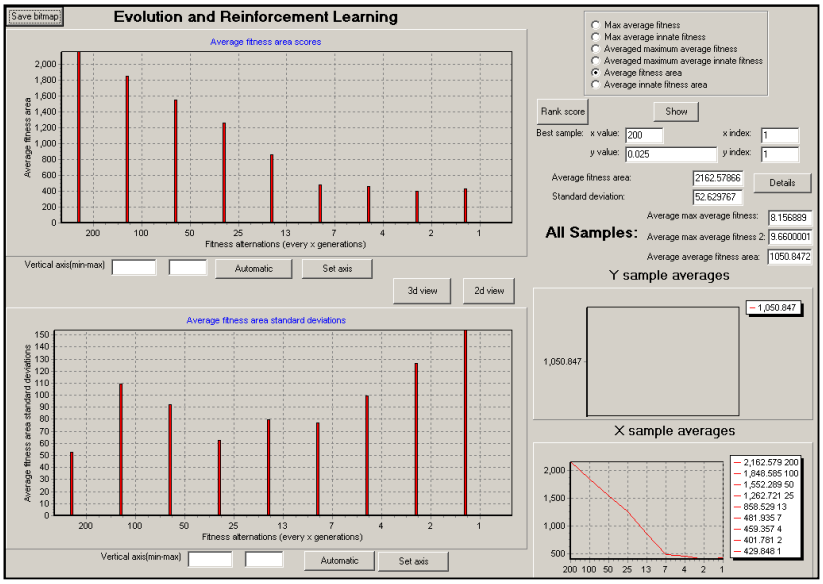


Figure P-9 Average fitness area scores produced by our Darwinian/Lamarckian global degree of neural network inheritance model in stable to highly unstable environments optimizing individual mutation rates with fixed reinforcement feedback values.

Evaluation of Our Darwinian/Lamarckian Global Degree of Neural Network Inheritance Model Optimizing Mutation Rates and Reinforcement Feedback Values in Stable to Highly Unstable Environments

The following data was produced using a fixed population mutation rate of 0.8 and optimizing individual mutation rates and reinforcement feedback starting from initial values of 3 for locating food, -3 for locating poison and –0.019 per action.

Maximum Average Fitness Scores

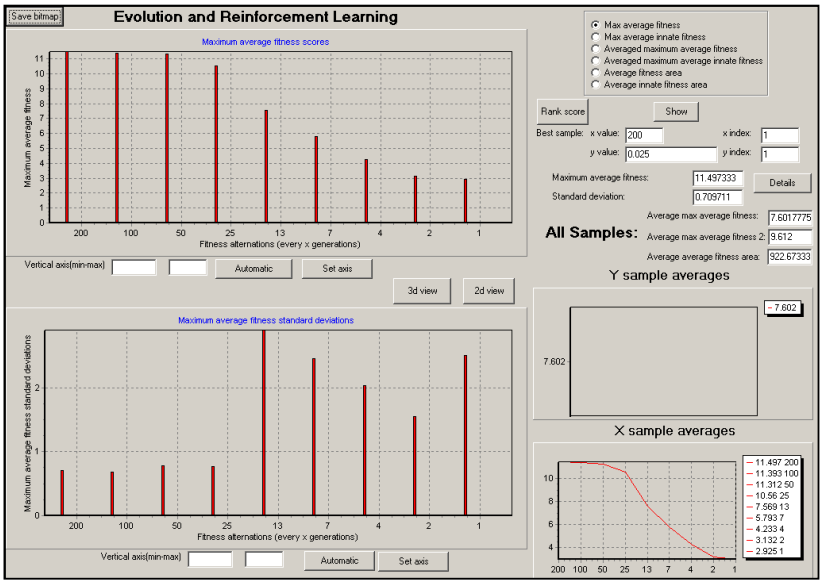


Figure P-10 Maximum average fitness scores produced by our Darwinian/Lamarckian global degree of neural network inheritance model in stable to highly unstable environments optimizing individual mutation rates and reinforcement feedback values.

Averaged Maximum Average Fitness Scores

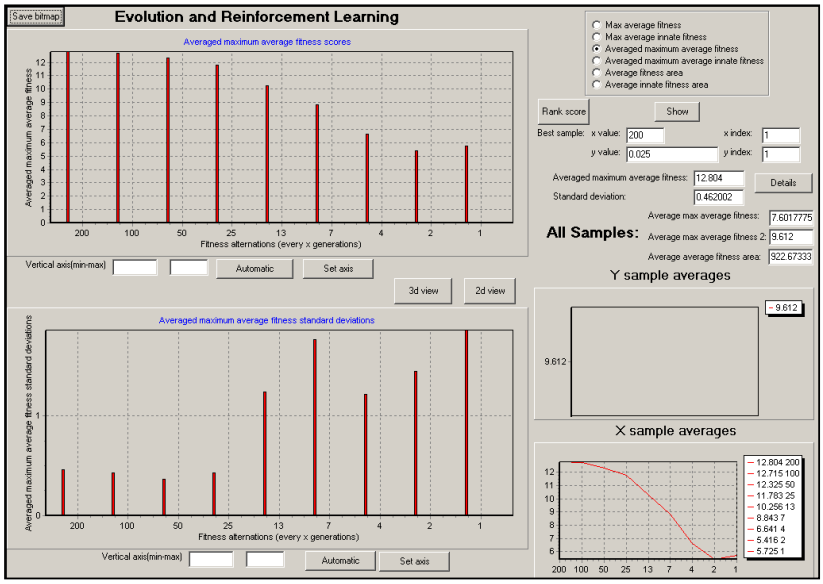


Figure P-11 Averaged maximum average fitness scores produced by our Darwinian/Lamarckian global degree of neural network inheritance model in stable to highly unstable environments optimizing individual mutation rates and reinforcement feedback values.

Average Fitness Area Scores

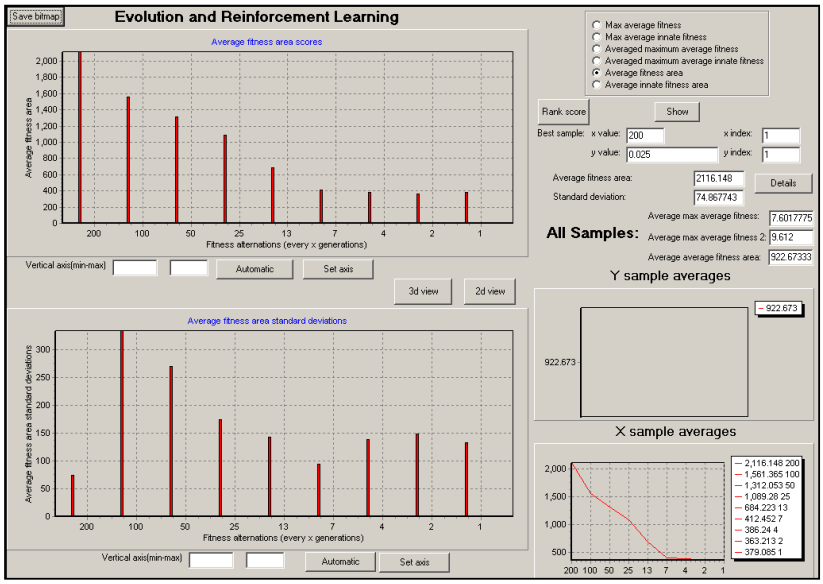


Figure P-12 Average fitness area scores produced by our Darwinian/Lamarckian global degree of neural network inheritance model in stable to highly unstable environments optimizing individual mutation rates and reinforcement feedback values.

Appendix Q

Evaluation of Our Darwinian/Lamarckian Local Degree of Connection Inheritance Model Utilizing Fixed Mutation Rates and Fixed Reinforcement Feedback Values in Stable to Highly Unstable Environments

The following data was produced using a fixed population mutation rate of 0.8 and fixed individual mutation rate of 0.1 with fixed reinforcement feedback values of 3 for locating food, -3 for locating poison and –0.019 per action.

Maximum Average Fitness Scores

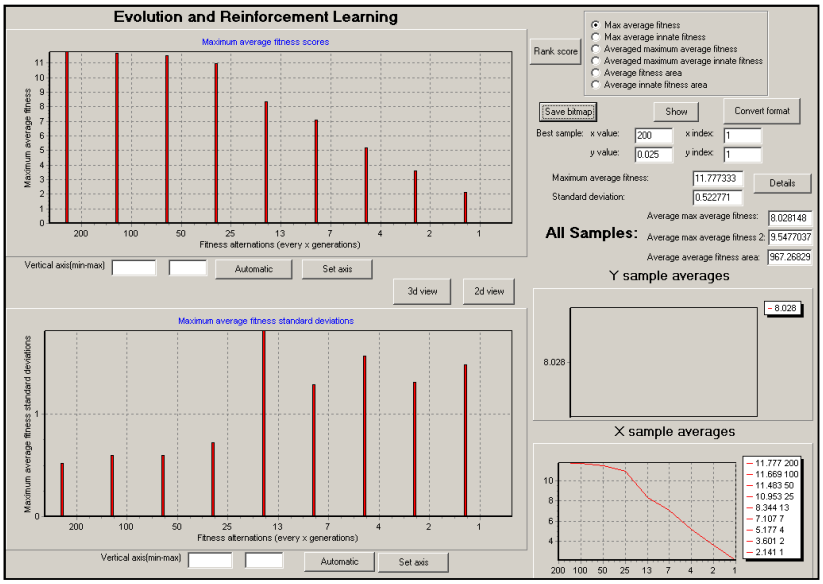


Figure Q-1 Maximum average fitness scores produced by our Darwinian/Lamarckian local degree of connection inheritance model in stable to highly unstable environments utilizing fixed mutation rates and fixed reinforcement feedback values.

Averaged Maximum Average Fitness Scores

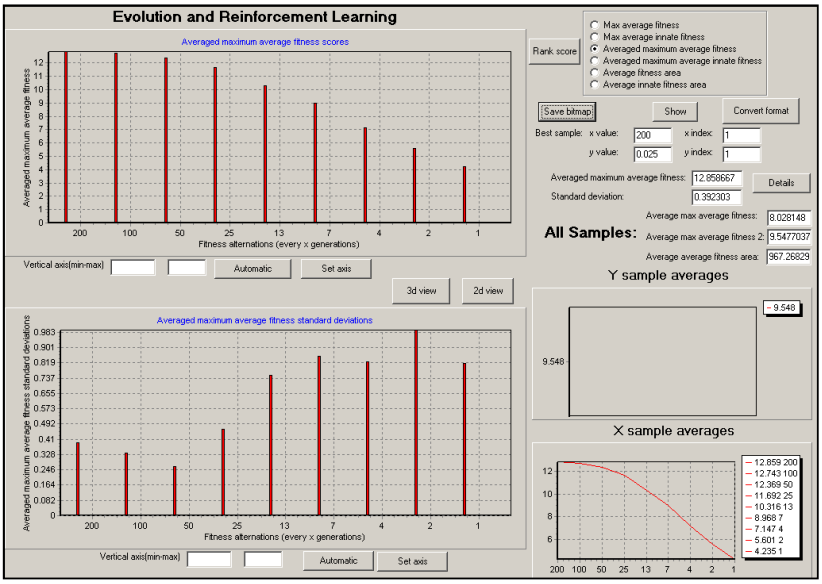


Figure Q-2 Averaged maximum average fitness scores produced by our Darwinian/Lamarckian local degree of connection inheritance model in stable to highly unstable environments utilizing fixed mutation rates and fixed reinforcement feedback values.

Average Fitness Area Scores

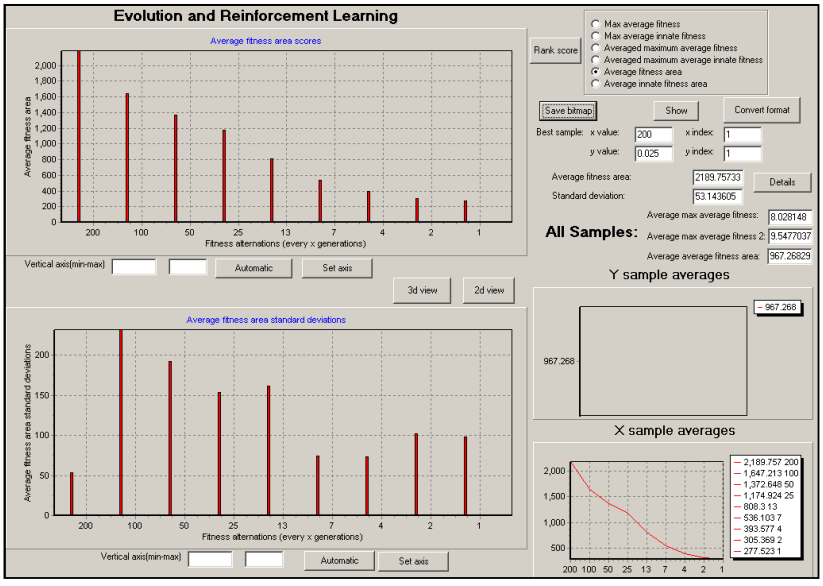


Figure Q-3 Average fitness area scores produced by our Darwinian/Lamarckian local degree of connection inheritance model in stable to highly unstable environments utilizing fixed mutation rates and fixed reinforcement feedback values.

Evaluation of Our Darwinian/Lamarckian Local Degree of Connection Inheritance Model Utilizing Fixed Mutation Rates and Optimizing Reinforcement Feedback Values in Stable to Highly Unstable Environments

The following data was produced using a fixed population mutation rate of 0.8 and fixed individual mutation rate of 0.1 and optimizing reinforcement feedback starting from initial values of 3 for locating food, -3 for locating poison and –0.019 per action.

Maximum Average Fitness Scores

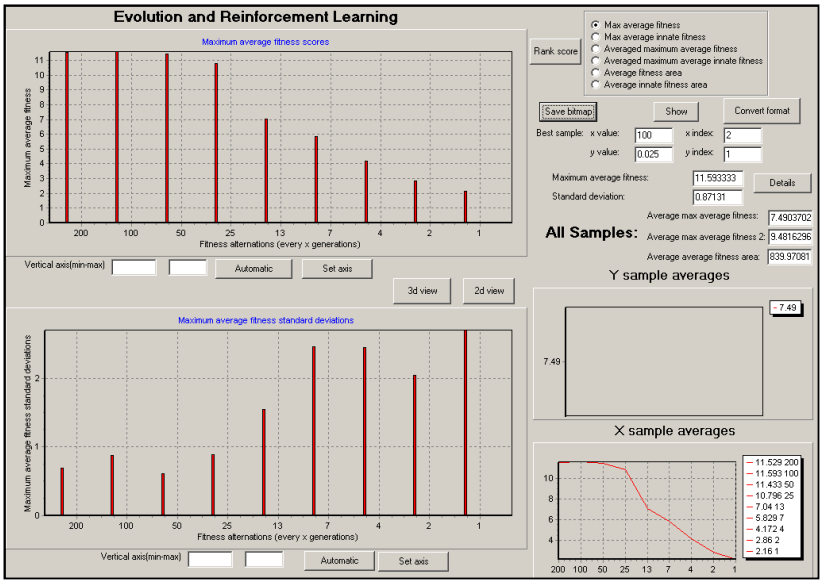


Figure Q-4 Maximum average fitness scores produced by our Darwinian/Lamarckian local degree of connection inheritance model in stable to highly unstable environments utilizing fixed mutation rates and optimizing reinforcement feedback values.

Averaged Maximum Average Fitness Scores

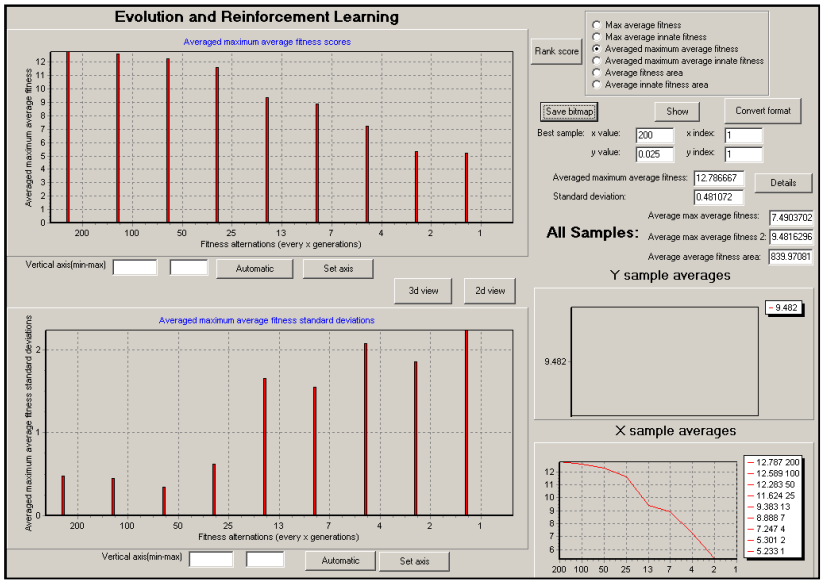


Figure Q-5 Averaged maximum average fitness scores produced by our Darwinian/Lamarckian local degree of connection inheritance model in stable to highly unstable environments utilizing fixed mutation rates and optimizing reinforcement feedback values.

Average Fitness Area Scores

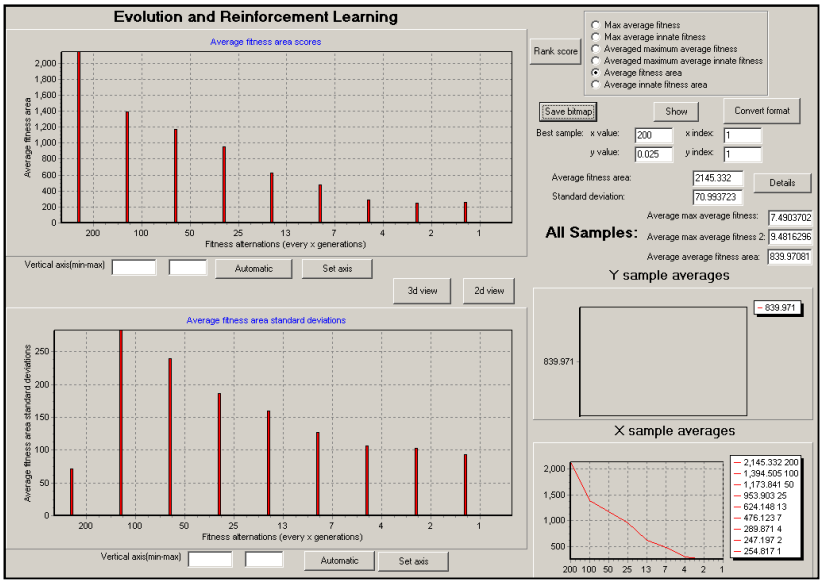


Figure Q-6 Average fitness area scores produced by our Darwinian/Lamarckian local degree of connection inheritance model in stable to highly unstable environments utilizing fixed mutation rates and optimizing reinforcement feedback values.

Evaluation of Our Darwinian/Lamarckian Local Degree of Connection Inheritance Model Optimizing Mutation Rates and Using Fixed Reinforcement Feedback Values in Stable to Highly Unstable Environments

The following data was produced using a fixed population mutation rate of 0.8 and optimizing individual mutation rates with fixed reinforcement feedback values of 3 for locating food, -3 for locating poison and –0.019 per action.

Maximum Average Fitness Scores

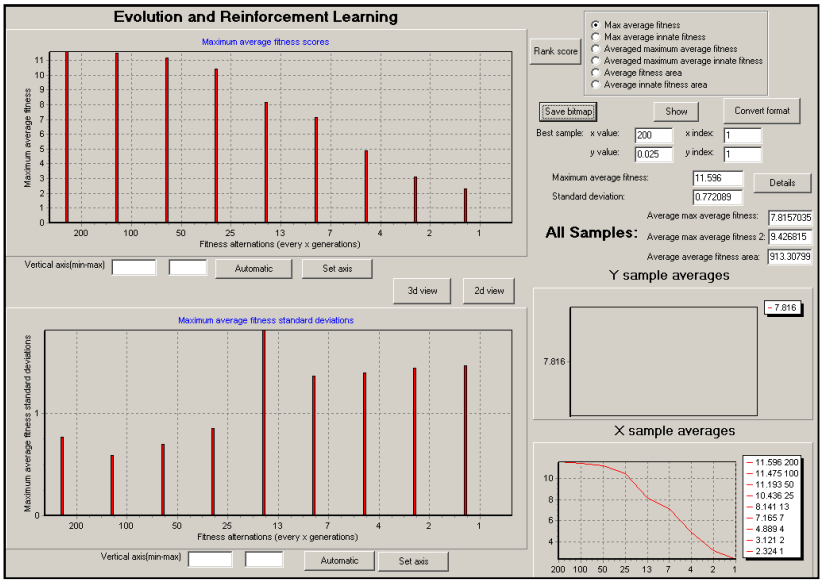


Figure Q-7 Maximum average fitness scores produced by our Darwinian/Lamarckian local degree of connection inheritance model in stable to highly unstable environments optimizing individual mutation rates with fixed reinforcement feedback values.

Averaged Maximum Average Fitness Scores

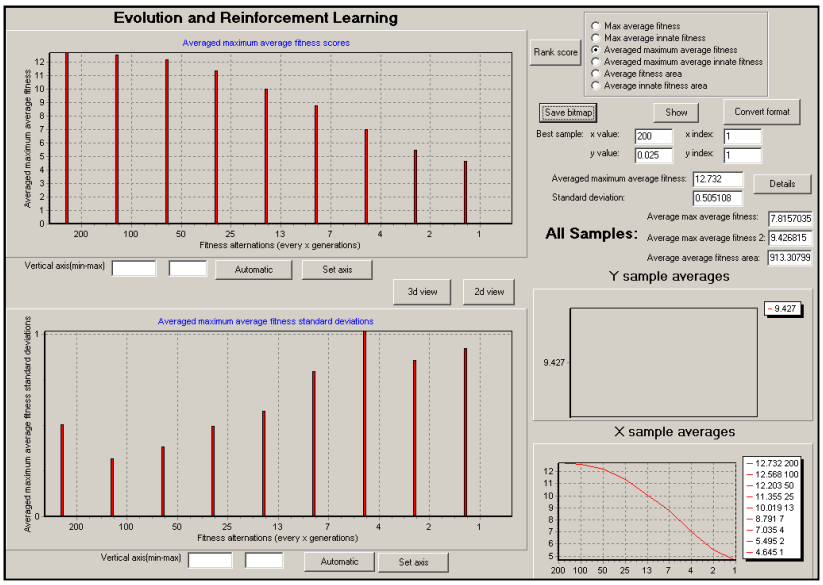


Figure Q-8 Averaged maximum average fitness scores produced by our Darwinian/Lamarckian local degree of connection inheritance model in stable to highly unstable environments optimizing individual mutation rates with fixed reinforcement feedback values.

Average Fitness Area Scores

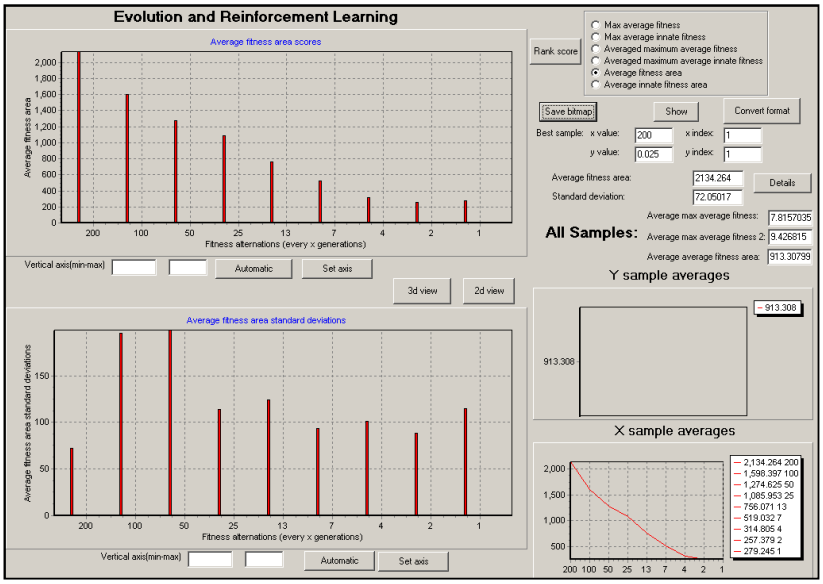


Figure Q-9 Average fitness area scores produced by our Darwinian/Lamarckian local degree of connection inheritance model in stable to highly unstable environments optimizing individual mutation rates with fixed reinforcement feedback values.

Evaluation of Our Darwinian/Lamarckian Local Degree of Connection Inheritance Model Optimizing Mutation Rates and Optimizing Reinforcement Feedback Values in Stable to Highly Unstable Environments

The following data was produced using a fixed population mutation rate of 0.8 and optimizing individual mutation rates and reinforcement feedback starting from initial values of 3 for locating food, -3 for locating poison and –0.019 per action.

Maximum Average Fitness Scores

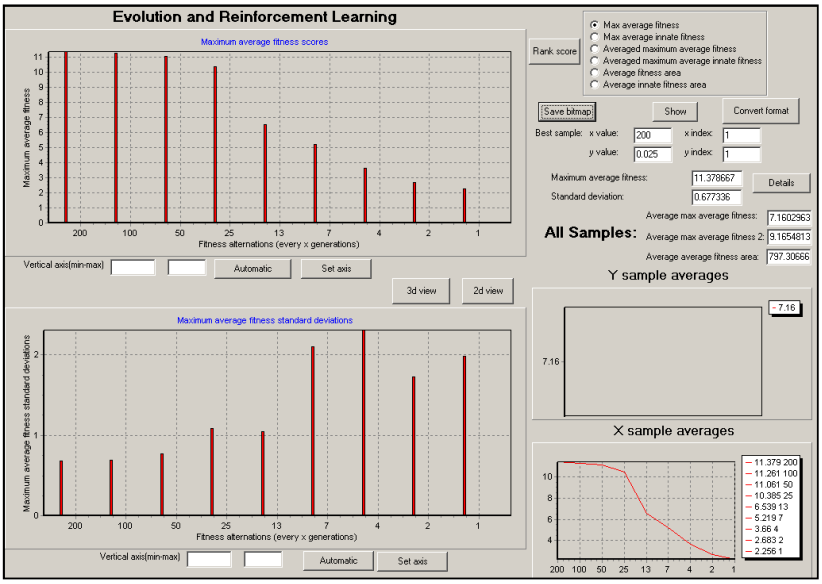


Figure Q-10 Maximum average fitness scores produced by our Darwinian/Lamarckian local degree of connection inheritance model in stable to highly unstable environments optimizing individual mutation rates and reinforcement feedback values.

Averaged Maximum Average Fitness Scores

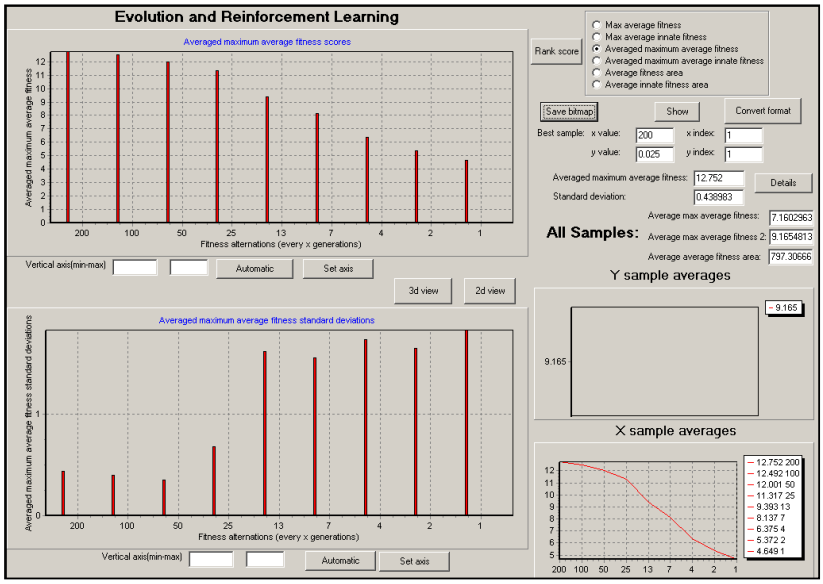


Figure Q-11 Averaged maximum average fitness scores produced by our Darwinian/Lamarckian local degree of connection inheritance model in stable to highly unstable environments optimizing individual mutation rates and reinforcement feedback values.

Average Fitness Area Scores

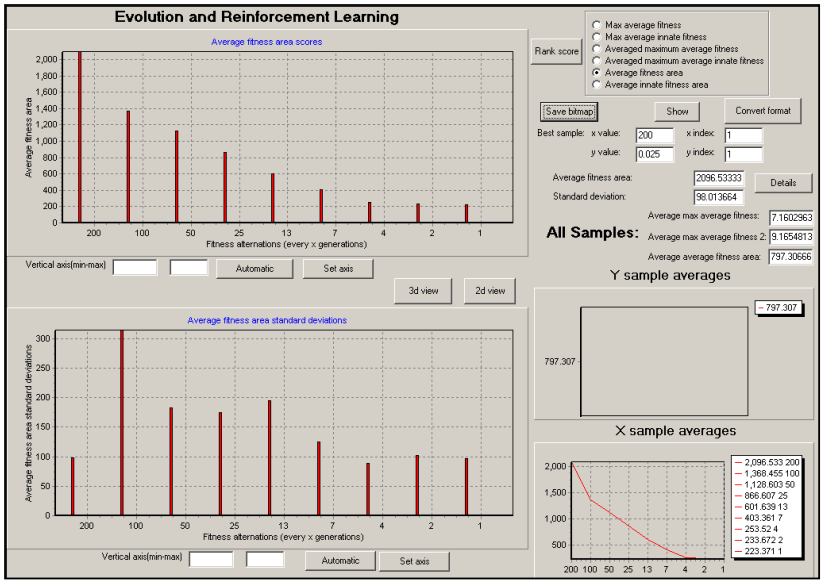


Figure Q-12 Average fitness area scores produced by our Darwinian/Lamarckian local degree of connection inheritance model in stable to highly unstable environments optimizing individual mutation rates and reinforcement feedback values.

Appendix R

Comparison Data

Maximum Average Fitness Scores

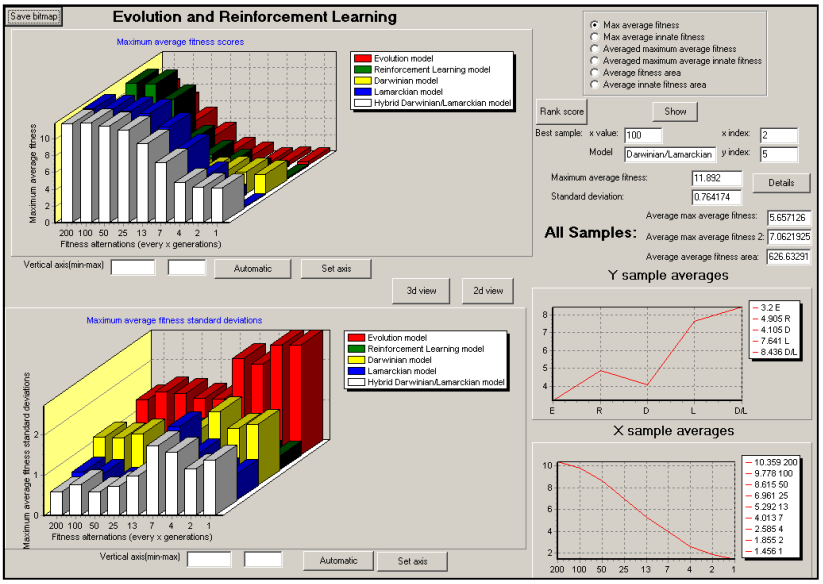


Figure R-1 3d perspective of maximum average fitness scores produced by the best performances of our evolution and reinforcement learning models in stable to highly unstable environments.

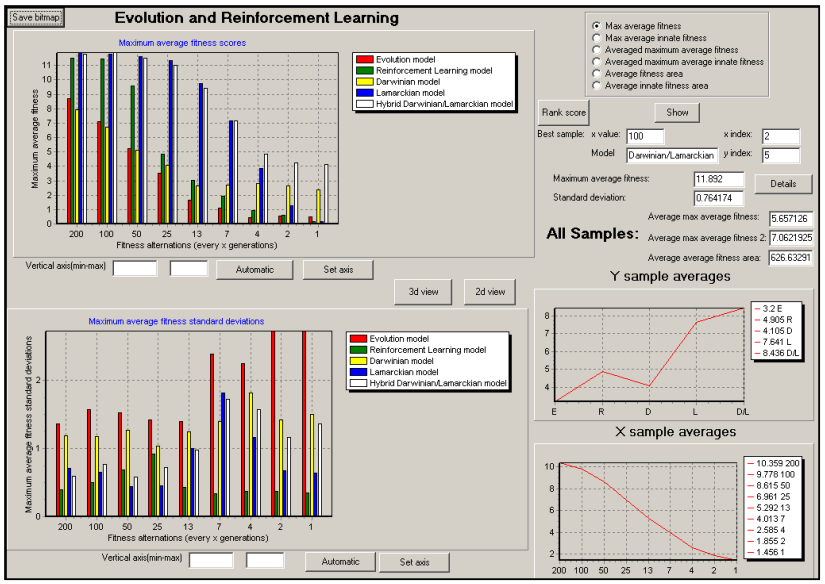


Figure Q-2 2d perspective of maximum average fitness scores produced by the best performances of our evolution and reinforcement learning models in stable to highly unstable environments.

Averaged Maximum Average Fitness Scores

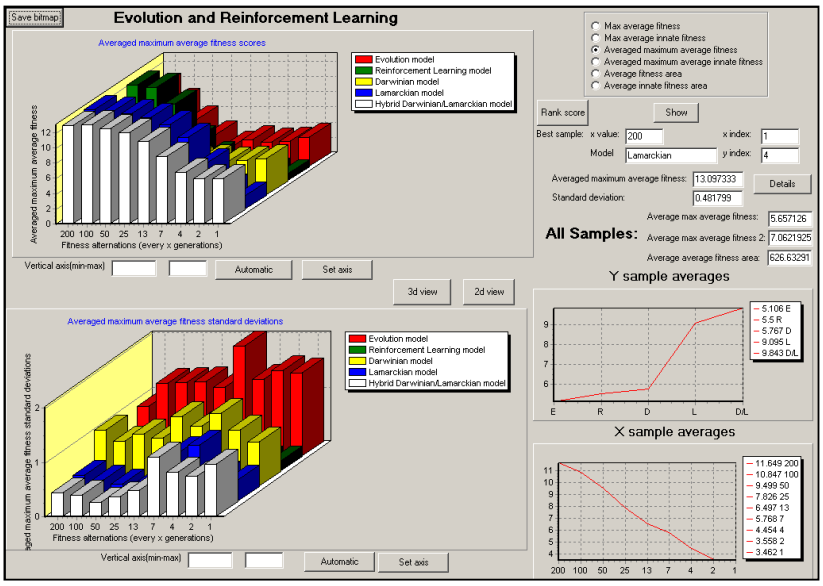


Figure R-3 3d perspective of averaged maximum average fitness scores produced by the best performances of our evolution and reinforcement learning models in stable to highly unstable environments.

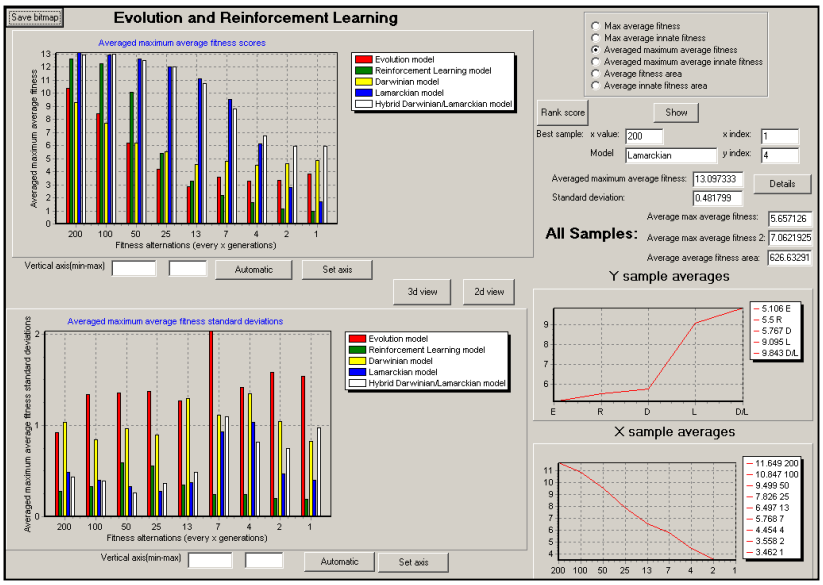


Figure R-4 2d perspective of averaged maximum average fitness scores produced by the best performances of our evolution and reinforcement learning models in stable to highly unstable environments.

Average Fitness Area Scores

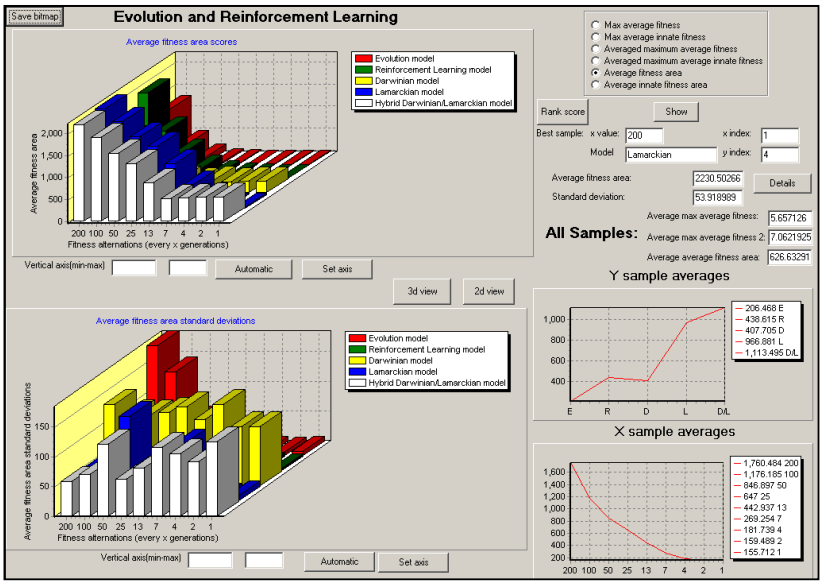


Figure R-5 3d perspective of average fitness area scores produced by the best performances of our evolution and reinforcement learning models in stable to highly unstable environments.

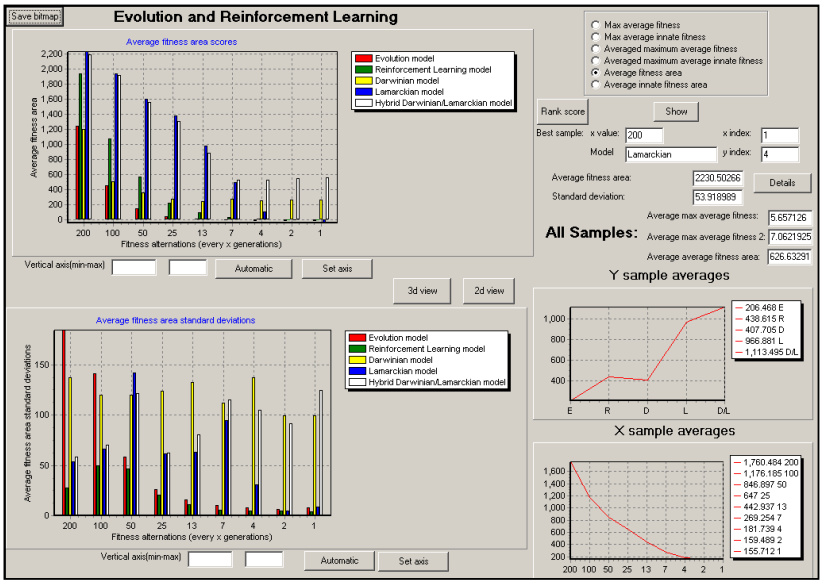


Figure R-6 2d perspective of average fitness area scores produced by the best performances of our evolution and reinforcement learning models in stable to highly unstable environments.