Neuroevolution of bipedal locomotion: algorithmic, balance penalty and morphological improvements for improved robustness and performance

Benjamin Jackson

Submitted for the degree of
Doctor of Philosophy
June 2022
Keele University

Acknowledgements

It shouldn't surprise anyone reading this who knows me to know that I am completely and utterly incapable of existing correctly without the constant unwavering support of multiple friends and colleagues. Here I will chronicle such efforts across the duration of the PhD, and then apologise later to the precisely one person that I have forgotten to mention. Said person, you have my genuine heartfelt thanks for what you've done and said in my support. Most of all, I couldn't have climbed this mountain without you. Thank you.

Firstly and fore-mostly, I have to thank my supervisor, Alastair Channon. I don't have enough fingers and toes to count the number of times this man has single-handedly saved this work from utter oblivion - be it through practical ways to approach the problems I faced that never would have occurred to me, or on occasion simply reminding me that I deserved to be in my position and not to feel so disheartened. His presence and experience throughout has been invaluable, and his attitude always left me with a smile on my face. Thank you Alastair, you gave me the chance of a lifetime. There's nothing wrong with people who ride horses, I swear.

Secondly, I need to thank my cohort and one year my senior, Ben Jolley. Making jokes in the labs during demonstrating, discussing politics on long journeys to conferences and irritating people around us -showing me how to do basic things I didn't know how to do. All of it helped me get the direction I needed, and reminded me to look on both sides of an issue. I can't thank you enough either. *My wife.*

Other people that deserve more credit than what I can write here include fellow PhD Khaled, for encouraging me to keep to an early rising schedule. Ian, our building's caretaker, for always showing me more kindness than my grumpy self deserved. Adam Stanton and James Borg, for

being the angel and devil figures of authority on my shoulders (although I won't say which is which). I would also like to thank my secondary supervisor, Charles Day, who provided an excellent point of alternate contact and was always ready with his head for a volley.

I'd also like to thank the KPA for its bountiful supply of alcohol (and the friends I've met there), as well as the Newcastle-under-Lyme Well-being Service. As a long-term sufferer of severe anxiety and panic attacks, they were a vital lifeline when I felt like I couldn't cope.

Finally, without the love and support from my family, I wouldn't even be in school, let alone an academic institution. Mum, Malcolm, Dad, Phat. Thank you from the very bottom of my heart for letting me achieve my dreams.

Abstract

Bipedalism is theorised to have emerged in humans in order to enable endurance running and tool use via the hands. It is one of the most complex styles of locomotion, with agents typically having a high center of mass and two feet very close together. It is therefore particularly difficult to build smooth cyclic motion on top of this instability.

Existing neuro-evolutionary methods for bipedalism involve the use of a central pattern generator or passive dynamics. A bipedal walker designed by Solomon *et al* was able to walk on rough terrain with a set of simple linear neural network controllers. I utilise control cost enhancements alongside additional elitism to initialise the walking agents in 3D without the 2D bootstrapping required by the original. The model is shown to be capable of walking in 2D and 3D, and of turning and walking toward a target point.

Improving another leading bipedal system, two fitness-function enhancements are applied to the 3D Humanoid-v1 walking task using a replica of the Salimans *et al* evolution strategy system. The first enhancement reduces control cost and the second enhancement penalises poor balance. Individually, each enhancement results in improved fitness and life-like gaits. Combining the two enhancements produces gaits that are more robust to noise in their actions according to statistical significance tests.

After producing single-turn behaviour in the previous work, agent fitness in the Solomon *et al* system is found to be improved according to a statistical significance test, by evolving agents alongside morphologies resembling a baby albatross. Pursuit-based turning behaviour is produced in the evolved albatross agents. The agents are required to pursue a target point as it moves further away and back and forth across the x-axis. This

task produces bipedal agents capable of making four consecutive turns in pursuit over a short time period.

Contents

List of Figures

Chapter 1

Introduction

1.1 Overview

Bipedalism is the locomotion of the human race and many other species in the biosphere. Theorised to have emerged in humans in order to enable endurance running and tool use via the hands, bipedalism crossed the threshold from a daily task to something closer to an instinct. Despite this, it is one of the most complex styles of locomotion. Walking on two legs requires constant balance to not fall over, as bipedal center of mass is normally high above the legs and the two feet in a bipedal agent are traditionally very close together. This results in high instability. It is therefore particularly difficult to build smooth cyclic motion on top of this to replicate bipedal motion. Other styles rely on extra limbs or wheels to ease the burden.

Perhaps the most influential work in evolved agent-based systems was produced by Karl Sims in 1994 [75]. By combining a genetic algorithm with a neural network, Sims was able to create evolved virtual 3D agents capable of several kinds of locomotion, from swimming to running to jumping.

However, one type of agent locomotion not featured in this work was bipedal walking. Evolved bipedal agents were largely absent from artificial evolution until Reil and Husbands published in 2002, evolving the parameters of a pattern generating control neural network to enable a virtual bipedal agent to walk [64]. In 2013, Solomon *et al* were able to use a set of simpler perceptron-based control neural networks to produce a bipedal agent capable of walking on rough terrain in 3D [77]. This represented a step forward in both robustness and network simplicity, although their most successful networks required 2D preliminary work before 3D locomotion could be achieved.

Another important aspect of agent based systems (particularly in bipeds) is morphology. By crafting agent morphology to assist in performing tasks, the neural network controllers' task can be simplified. This is often seen with soft body parts or passive joints. One of the earliest bipedal examples of this, McGeer designed a walking robot with no control that was capable of walking down slopes. It used only gravity to swing its legs forward and roll itself over on its curved feet [51]. This passive leg swinging was very similar to the way humans swung their legs below the knee as they walked, and would lead to many more studies into the utilisation of morphology to aid control.

Most recently, Heess *et al* have utilised deep reinforcement learning algorithms, instead of genetic algorithms, to achieve high quality complex locomotion [35]. Their bipedal agents were able to run, walk, jump and even slalom around obstacles with only the basic reward from travelling forwards, in increasingly complex environments. This behavioural complexity is almost unheard of in bipeds utilising genetic algorithms.

As such, there is ample motivation to fill this void. Studying bipedal agents is important as bipedal agents exhibiting more complex behaviour in simulation could be adapted for real world stability testing under various dangerous circumstances. These bipedal agents could then traverse unstable buildings or dangerous environments in the place of human agents. This could therefore lead to a complete reduction of fatalities amongst rescuers and other emergency workers. Another reason to study bipedal agents is to produce responsive behaviour in real world care assistant robots. These bipedal agents could assist the elderly whilst making them feel more relaxed around an agent with a humanoid shape. This could then lead to a greater quality of life amongst the residents of care homes and other assistance facilities.

But why choose to use evolutionary systems when deep reinforcement learning algorithms have already achieved a better result? The first benefit of an evolutionary method is the biological aspect of the methodology. Evolutionary processes being modelled after evolution in the biosphere allows them to provide a trajectory of behaviours leading up to the final goal. This can potentially teach us more about our own evolutionary trajectory or even other locomotive pathways not chosen in our biosphere. The other benefit of an evolutionary method is that they are able to increment through complex tasks. Compared to reinforcement learning, which can suffer from over-fitting and a lack of generalisation, evolutionary processes are much easier

to increment through multi-faceted tasks due to their mapping of task performance to a fitness function and a genotype. It is possible to apply a new fitness function for a new aspect of a task starting with a genotype already preloaded from the previous task, something reinforcement learning can again struggle with due to over-fitting. This could lead to potentially more complex behaviours in simulated agents in the long term via smaller increments, if evolution can overcome the initial hurdles.

1.2 Aims

- **Increase Walker Fitness.** I feel the first step to produce more complex behaviour in bipedal agents is to improve the existing walking fitness for robustness, behaviour and stability.
- **Utilise Morphology to Simplify Bipedal Task.** Bipedal Walking requires balance at all times by definition, making for a difficult task. Many systems have seen success from utilising morphology in order to make complexity easier to achieve. This could be utilised for the difficulties of bipedal walking task.
- **Produce Complex Behaviour.** Many evolutionary bipedal systems are successful with compass walks (walking solely from leg swinging, without knee motions) and other simplistic gaits but do not feature any behaviour beyond forward walking. Producing something beyond this with a genetic algorithm represents a step forward toward the complexity seen in other types of systems.
- **Highlight the merits of evolutionary processes for bipedal walking.** I can also highlight some of the perks for choosing evolutionary systems over other methods. These perks include incremental evolution and the biological insight evolutionary systems can provide.

1.3 Structure of Thesis

This thesis has nine component Chapters, the content of which are as follows:

- **Chapter 1**, this Chapter, introduces the thesis, outlines my overall aims, and describes the content of each Chapter.
- **Chapter 2** covers the background of agent based evolutionary systems, from the advent of the genetic algorithm and neural network, to Karl Sims' virtual creatures and through to modern agent based systems.

- **Chapter 3** focuses on the origins of bipedalism and modern bipedal systems. I then analyse existing evolutionary bipedal systems in more depth, noting their results and limitations.

- **Chapter 4** represents the first novel work of the thesis. Here I aimed to enhance fitness and produce complex behaviour. I enhance the Solomon *et al* Linear Reactive bipedal system [77] using control enhancements and increased elitism, enabling direct initialisation into a 3D physics engine, without the required prior 2D physics engine bootstrapping. I produce stable walking gaits in 2D and 3D. I also produce single-turn behaviour in 3D, in a similar fashion to Reil *et al* [64].

- **Chapter 5** represents the second novel work of the thesis. Here I aimed to again enhance fitness and produce complex behaviour. I continue the theme of optimising leading bipedal systems from the previous Chapter to demonstrate the robustness of my methods in a modern context. I introduce control constraints again, this time to a deep evolutionary bipedal system [68] to produce both fitter gaits and gaits that feature less shuffling behaviour. This work was published in the proceedings of the ALIFE 2019 Conference.

- **Chapter 6** is the third novel work of the thesis. Here I aimed to enhance fitness through morphology. I revisit the enhanced system from Chapter 4 as its simple neural network controller is more sensitive to morphological change. I use morphologies based on a baby albatross and the balance and control cost enhancements from Chapter 5, to produce bipedal agents with drastically higher fitness. This work was published in the proceedings of the 2020 IEEE Symposium Series on Computational Intelligence.

- **Chapter 7** is the fourth novel work of the thesis. Here I aim to produce further complex behaviour. I utilise the most successful albatross morphology from Chapter 6 to produce complex behaviour on a new pursuit-based turning task. Following on from the turning in Chapter 4, I produce agents capable of turning multiple times across the x-axis whilst following a moving target point, in a slalom-like way. This is to the best of my knowledge, a unique behaviour amongst agents with evolved neural network controllers.

- **Chapter 8** is the final novel work of the thesis. Here I aim to demonstrate the merits of evolutionary techniques. Used previously by Solomon *et al* and

Stanton *et al*, I attempt to utilise incremental evolution to improve the number of moves of the turning agents produced in Chapter 7. No improvements in the number of moves are produced.

- **Chapter 9** summarises all contributions to the field across the five novel works in relation to my aims, and discusses conclusions and future work that could be undertaken as a result of findings.

Chapter 2

Background - Virtual Agents

2.1 An Introduction to Neuroevolution

Within the context of this field, Neuroevolution refers to the process in which a neural network (also referred to here as a neural network controller and a neural control network) is evolved to control an agent in order to achieve a goal or task. In this thesis, I study the neuroevolution of virtual agents. A virtual (sometimes referred to as simulated) agent is defined as an agent or other actor which is simulated within a physics engine instead of the real world. I chose to study the neuroevolution of bipedal locomotion in virtual agents instead of real world (sometimes referred to as physical) agents due to the costs required to build physical agents. A physical bipedal robot would require building and material costs for each morphology and environment - simulations allow for instant building and easy adjustment for only the price of the physics engine. In addition, virtual agents can be simulated at speeds much faster than the physical world. This allows experiments to be completed at a much faster rate than physical agents.

A neural network is made from nodes, each featuring a weight for each other node they are connected to. These weights modify values from input to output, with each node summing its inputs multiplied by their weight values, then applying a function to produce its output. In virtual agents, network inputs are typically agent sensors and network outputs are typically agent actuators. Learning algorithms such as Back-Propogation [66] and Reinforcement Learning [88] are often applied to these to learn a desired behaviour over a creature's lifetime. In the case of neuroevolution, the network weights are instead modified via an evolutionary process such as a Genetic Algorithm or an Evolution Strategy, to produce the desired input/output mapping. This control mapping will then enable the agent to produce behaviour achieving its

goal.

Holland conceived Genetic Algorithms in 1962 as a way of mathematically modelling the process of biological evolution [37]. Rechenberg *et al* created a similar methodology in 1965 dubbed "Evolution Strategies" to optimise airfoil parameters in a one-parent one-offspring system [63]. Fogel *et al* also developed an evolutionary programming technique through mutating finite state machines around the same time [30]. As mentioned previously, evolutionary processes involve a genotype, mimicking the DNA of a subject. This is usually a fundamental aspect of the subject, such as neural network weights. Evolutionary processes also involve a fitness measurement, related to the goal. A population of subjects is tested and their fitness is measured. The most successful subjects are kept for the next generation. This next generation is produced with new subjects whose genotypes are randomly mutated from those of the previous generation. In this way useful mutations accumulate over many generations, increasing fitness until the desired goal is reached.

There are other evolutionary measures besides fitness. Lehman *et al* devised novelty search [43]. This involved selecting creatures based on their uniqueness compared to others. Barnett *et al* worked on neutral networks. These were genotypes close to that of the subject that had identical fitness. They exploited these to reach higher fitness peaks [3]. Wilke *et al* also studied an asexual population capable only of negative and neutral mutations [92]. They discovered that evolutionary dynamics across a neutral space happen in a much more clustered manner, as opposed to a mutation spreading through the population's genomes gradually.

Diversity is important to evolutionary processes. Mouret *et al* demonstrate this using multi-objective fitness and fitness sharing techniques to enhance population diversity [57]. This resulted in considerable improvements in several systems. This was also shown by Wagner *et al*, who complexified the genotypes of agents to allow for more diversity [87]. This resulted in either improved fitness or new emergent behaviour across all tests. Nygaard *et al* used multi-objective optimisation to achieve a variety of high fitness quadrupedal locomotion strategies in a legged robot over a short time [60]. Berland *et al* designed a cheap spiking neural model for agent-based models, which iterated through a set of states for agents with longer time-steps [5]. This was computationally cheaper than standard spiking models, and capable of modelling real chemotaxic behaviour with only three neurons.

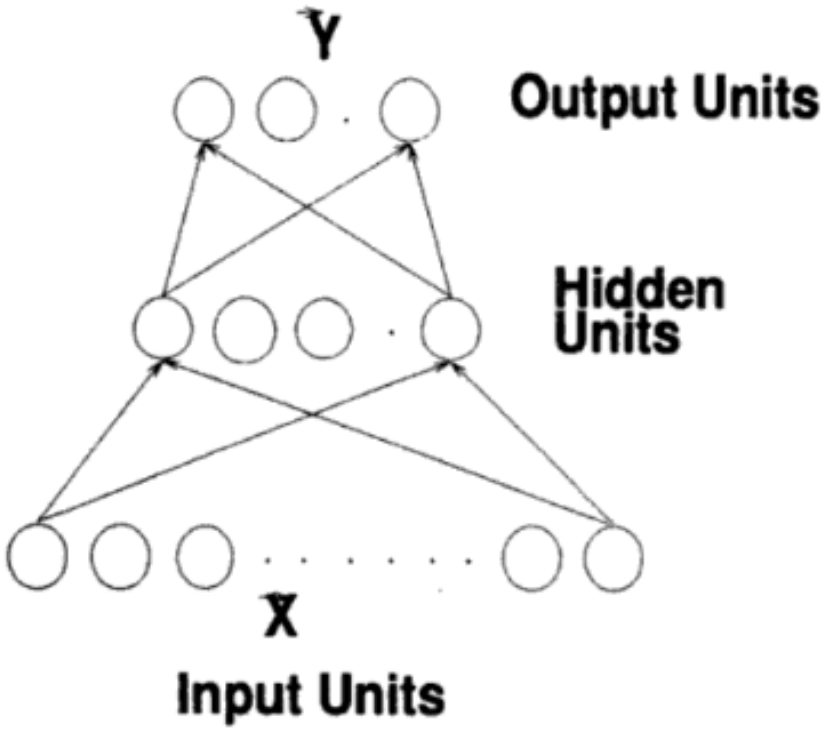

Figure 2.1: **An example neural network layout.** Input is processed from the inputs, through the hidden units to the output. Each node applies an activation function to the sum of its inputs multiplied by their respective weights, to produce an output. [66]

Mutation rate can also be a significant factor in the emergence of agent based behaviour. Grefenstette utilised a second "meta" Genetic Algorithm to adjust the population size, mutation rate, crossover rate and other features of a genetic algorithm [33]. This produced high fitness levels on a set of various optimisation tasks. Ochoa *et al* investigated the relationship between error thresholds and mutation rates [61]. An error threshold in an evolutionary process is defined as a critical mutation rate beyond which genotypes mutated by an evolutionary process lose their acquired behaviour more frequently than retaining it. They demonstrated that the optimal mutation rate lay just below the error threshold. Uyar *et al* varied mutation rate based on genotype [83]. They split genotypes into two groups, each having their own separate mutation rate. They then updated each mutation rate based on success. This outperformed similar approaches on the "one max" problem (maximising the number of 1s in a string). Cervantes *et al* altered their mutation rate by different subsections of the population, mutating each agent by fitness rank [14]. They first performed crossover of nearby ranked agents to combine dissimilar solutions, then reevaluated the population, applied the proportional mutation and selected for fitness. This was more effective than a standard genetic algorithm across multiple types of fitness landscape.

One of the earliest 3D neuroevolution examples was Sims' virtual creatures [75]. By combining a simple physics engine, a neural network and a genetic algorithm, Sims showed it was possible to evolve 3D agents to achieve a variety of real, life-like behaviours. Sims used a graph-based genotype to form 3D agents. Each node in the graph represented both a polygon limb of a creatures body and a set of neurons to connect to the neural control network. The graph nodes also contained morphological properties such as connection position, dimensions, number of connections, reflection and terminality. The networks formed a structure travelling from the agents sensors through to their joint actuators. A set of global neurons disconnected from any polygon were also granted to each agent, such that it may utilise them for global control purposes. Sims evolved a variety of behaviours including swimming, jumping and light following, with multiple strategies for each behaviour.

Co-evolution is another important evolutionary concept, often observed in nature between competing species [28]. These species will find themselves adapting over time to each other. This is most commonly seen in predator-prey relationships. Sims would go on to investigate co-evolution amongst his creatures [74]. Two creatures

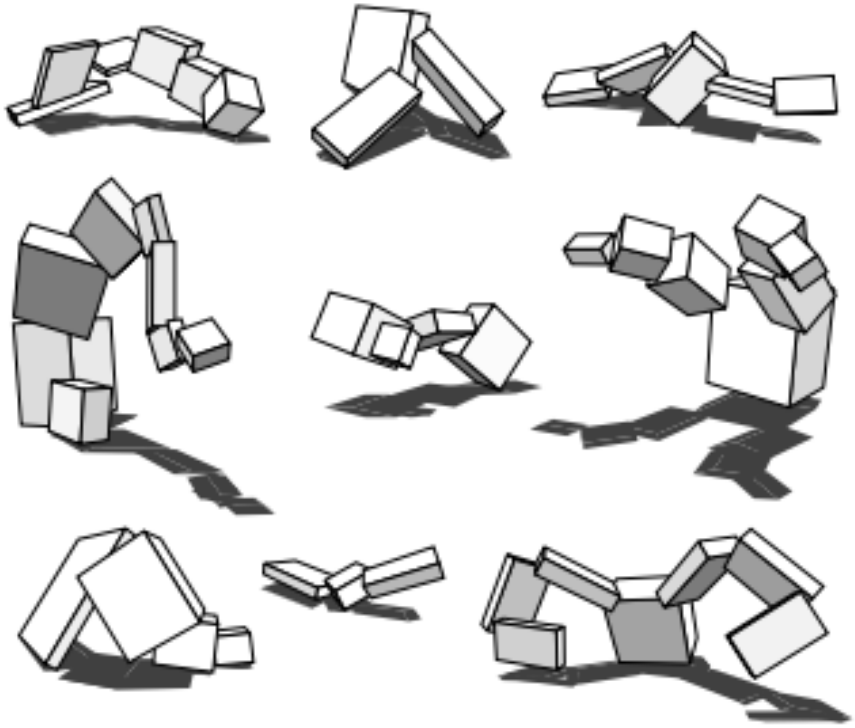

Figure 2.2: **Sims virtual creatures.** Sims evolved these creatures from a combined network/morphology genotype using a genetic algorithm. He produced running, jumping, swimming and light following behaviours [75].

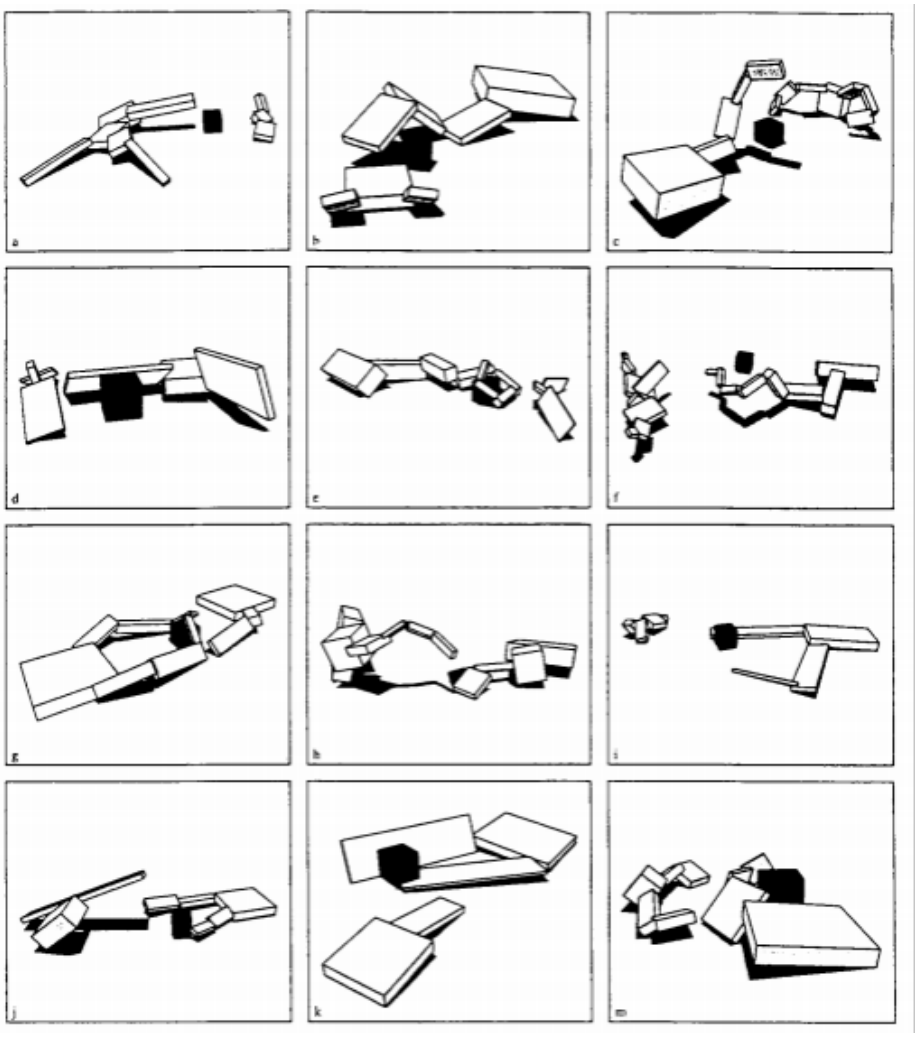

Figure 2.3: **Co-evolved virtual creatures.** Sims evolved his creatures against each other for a control task. Seen here are different strategies adopted by the competing virtual creatures to maintain control of the black cube. [74]

would compete to see who could gain control of a cube placed between them. This produced an array of intelligent, complex strategies that emerged from evolutionary interactions between the two species. They exploited each others' weaknesses and produced counter strategies, in a co-evolutionary arms race. Watson *et al* would later show competition between opposing species boosted diversity and overall fitness further than was possible alone [89].

2.2 Modern Agents

In early instances of learning systems, often the methodology chosen was to increase the complexity of the controlling process. Waxman *et al* produced a vehicular agent which made all decisions based on its own internal plan, which it had to continually update over time in order to be reactive and responsive [90]. This methodology initially showed good results, such as Movarec *et al* [56], but was very computationally intensive, and the agents struggled to keep up with more powerful models. This viewpoint changed when Brooks demonstrated the power of simple reactive control processes, with little to no memory [12]. With these, it was possible to create agents that achieved equally complex behaviours using the interactions between the control process and their environment alone. Brooks modularised this control scheme, using an override system to achieve complex behaviour in robots. He dubbed this the subsumption architecture.

Following this, bio-inspired embodied design experienced a huge surge in popularity, both in robots and in silica. Some examples of this included Briod *et al*, who built a flying robot utilising insect collision-recovery behaviours [11]. Calisti *et al* tested octopus based locomotion and grasping behaviour by utilising soft limb materials [13]. Vaughan *et al* designed a hexapod robot for rough terrain and lunar exploration based on a stick insect [85]. Ijspeert *et al* designed a salamander-based robot that was capable of transitioning between walking and swimming gaits based on a spinal cord model [39]. Huang *et al* evolved a neural network for a robotic hand to grasp previously unseen objects [38]. Lipson *et al* evolved virtual agents for locomotion and processed them into agents for an underwater environment [47]. Chaumont *et al* evolved block-throwing creatures using Sims' methodology, discovering a variety of successful strategies [16]. Miconi *et al* also adapted Sims' work, adding force and damage parameters to competing agents and observing a variety of novel strategies [53]. Crespi *et al* produced an underwater fish-inspired robot with multiple gaits, able to avoid obstacles whilst going backwards [22]. Bharaj *et al* utilised evolutionary optimisation to refine personalised automata created through a drag and drop interface, such that they were capable of locomotion, for both bipedal and quadrupedal designs [6]. Whilst these evolved automata were capable of bipedal gaits across several customised morphologies, they lacked any form of neural control network, using linear motors instead. Cully *et al* evolved several unique neural network controllers for a hexapod robot to reach a point, as opposed to a singular complex neural network

controller [23].

Bongard *et al* designed a virtual robot arm, introducing it to increasingly complex object manipulation tasks [8]. A simple CTRNN, or continuous time recurrent neural network, controlled the arm. This style of network features recurrent backward flowing connections that are applied through multiple timesteps. They varied the number of fingers the arm used between three and five. They then gave the arm to a series of objectives to learn, including grasping an object, lifting an object as high as possible, and classifying different types of objects. To increase complexity further, they forced the arm to learn several tasks at once. They tested the arm's ability to perform over seven regimes of task complexity, from each task on its own to all three tasks at once. With a single objective, they saw no increased performance due to cheating strategies. With two, they began to see a more noticeable increase in performance. All three objectives together showed the most improvement overall. They were able to determine the different combinations of morphology parameters that allowed for improvement on each hand type. They also noted that evolving the morphology never made fitness worse, and the more they allowed it to evolve, the more robust their behaviours became. They hypothesised that if this improvement was due to poor starting conditions then evolution would have taken it much further from its starting point. Instead of which, the arm only changed slightly from its default morphology.

Corucci *et al* investigated the use of on-line morphological changes for a soft underwater octopus-like robot [21]. They accommodated walking, running, and swimming gaits. This allowed easier gait transitions. They first evolved a population of simulated robots using either a standard or novelty search based genetic algorithm. After this they clustered individuals in their population based on genotypes and morphological parameters (angle of buoyant, buoyancy, crank rotation). They then selected one agent from each cluster to apply morphing. It was shown that the novelty-based strategies produced much more varied behaviour, whereas the standard genetic algorithm spent the experiment optimising a single strategy. They produced walking, running and sculling behaviours.

Lessin *et al* produced an evolved fight or flight response, using a system they dubbed the "ESP system" with morphology and neural network controllers almost identical to those used by Sims [44]. Differences included a newer physics engine and

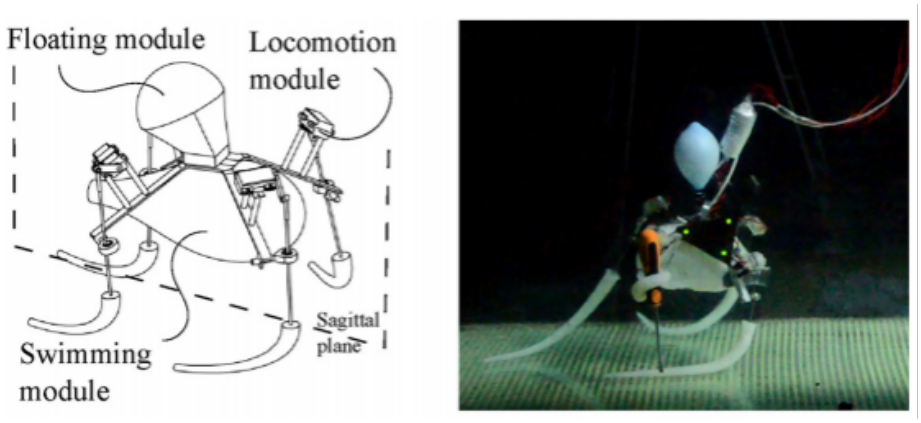

Figure 2.4: **The soft octopus robot.** Developed by Corucci *et al*, their robot featured flexible feet and a floating module [13]. Their robot evolved walking, running and sculling behaviours, and was able to morph between them using on-line morphological changes.

the usage of spring-like tendons to move limbs. Their ESP system consisted of three ideas. Firstly, "syllabus" - creating an ordered sequence of fitness goals. Second, "encapsulation" - grouping neural control network nodes into behaviour sections for modularisation. Finally, "pandemonium" - allowing the behaviour with the most active neurons to dominate, preventing clashes. They were able to achieve a considerable number of Sims' behaviours through ESP, including forward locomotion, reaching high, and moving toward a light source. They were also able to achieve striking (jumping on a target), attacking (moving to a light source combined with striking), turning away from a light, and retreating (combining forward locomotion and turning away from the light). Their final and most impressive result was the fight-or-flight response. This behaviour was a combination of attacking and retreating based on the amount of light given off in the nearby environment (larger agents gave off more light). They would later extend their ESP methodology to evolve morphological adaptations per behaviour [45]. The resulting population of agents had a much more diverse set of strategies for striking and high reaching skills through increased morphological adaptation.

Stanton *et al* [80] produced a quadrupedal agent-based system to solve the River Crossing task [65]. The task required agents to collect rocks to build a bridge across a river, whilst avoiding traps. In order to place the rocks, the agent must learn a searching strategy, but to avoid the traps at the same time, the agent must balance its search with caution. To circumnavigate this contrast, they created a hybrid net-

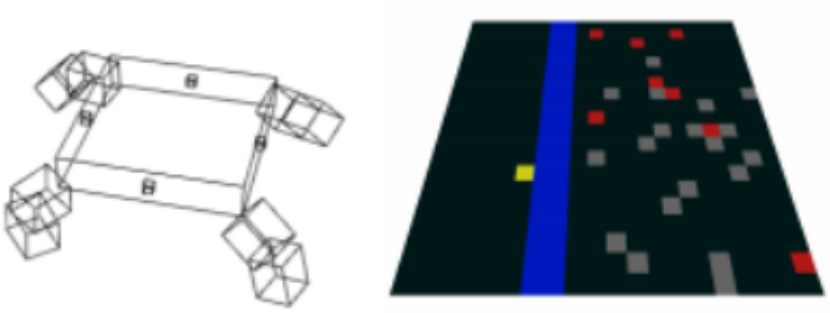

Figure 2.5: **Stanton's agents (left) and the river crossing task environment (right).** The agents' neural network controllers were evolved to move the grey stones to build a bridge across the river to the goal, whilst avoiding the red traps [80].

work topology. Their topology had four main components. The first component was the Decision Network, a feed-forward neural network with neurons responsible for choosing the agent's responses to inputs, from which the world information was fed into. This led into the next section of the network, the Shunting Model, which was a grid network with each node representing a cell in the task environment, building a map for the agent to traverse. The world information also fed into this network. This led to the Physical Network, featuring a single feed forward hidden layer, converting the grid information into desired values. This network also took in walker sensor information. Finally, the pattern generator section applied an oscillatory function to obtain rhythmic motor outputs from the desired values. The agents were evolved incrementally through six levels of the task. The first two stages featured basic locomotion and resource gathering. The final four stages were then various increments of the river crossing task, from no river to rivers of width 1, 2 and 4 cells. 65% of all evolved agent populations were able to produce an agent which could complete all the iterations of the task. This subset of the populations also exhibited the largest strategic diversity during initial iterations.

Incremental evolution is often used in evolutionary systems to reduce the difficulty of learning complex tasks [17]. Incremental strategies involve evolving a system through several sub-tasks one after the other, using the winners from the previous task as a starting point for the next. One of the main issues in neuroevolution is the bootstrapping problem. The bootstrapping problem refers to the difficulty of evolving multifaceted results from scratch without guidance. The more complex the task, the more likely the system is to be initialised far away from an optimum in

the fitness landscape. This increases the likelihood of the system becoming trapped in fitness plateaus. Incremental evolution can be used to avoid this problem and traverse the landscape via sub-tasks. Gomez *et al* used it to evolve prey-capturing and pole balancing behaviour via an adaptive network topology [32]. They were able to produce more successful behaviours using the technique and speculated that the technique could be used for many other agent-based tasks, provided they featured a clear hierarchy of sub-tasks from simple to complex. Mouret *et al* used incremental evolution to generate target-following neural network controllers in winged agents [58]. They used pattern generator networks to maintain a constant speed and altitude, first evolving linear flight and then incrementally evolving pursuit.

A more recent investigation of incremental evolution, Stanton *et al* evolved a quadrupedal agent to climb over a range of wall heights [79]. They incrementally evolved their agents through increasingly complex wall height configurations. They equipped the agents with a simple neural network translating their location and orientation into output torques, and evolved their weights using Harvey's SAGA algorithm [34]. They tested different complexification strategies with incremental evolution. Homogeneous strategies consisted of static and linear functions of wall height, whilst heterogeneous strategies featured oscillatory and non-linear functions. They were able to show that even the weakest heterogeneous strategy, randomly increasing the height of the wall, was able to produce better results than the best homogeneous strategy, the linear increase. This linear increase strategy was initially able to keep up with the heterogeneous strategies, but degraded over many iterations, showing signs of catastrophic forgetting - forgetting one task when learning another. The oscillatory heterogeneous strategies were the most successful, finding optimal frequencies to obtain near-perfect success rates.

Possibly the biggest advance in neural network controllers of the last decade was a recent explosion of their deep application. Krizhevsky *et al* trained a deep convolutional neural network (i.e. one with thousands of parameters) to classify more than a million images into a database of 1000 different classes, with error rates as low as 37.5% [42]. This level of error rate in image classification was as of then unprecedented, as was large scale image processing of that scale. To achieve this, they firstly converted the RGB input images to grayscale, before feeding them into the network. Their network consisted of five convolutional layers, each connected through a pooling layer. The convolutional layers functioned via applying a number of kernel grids to

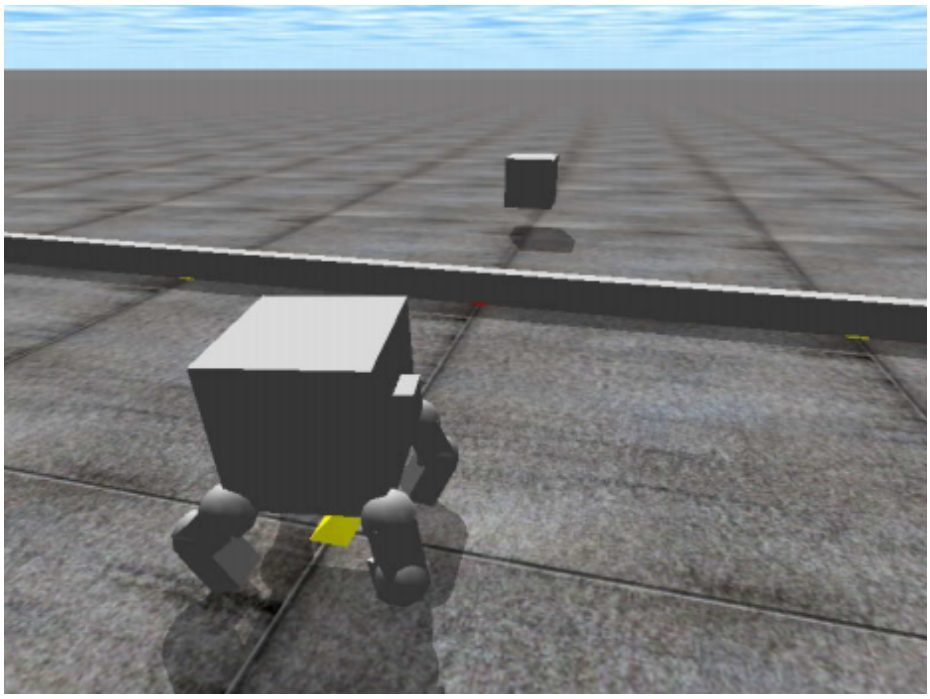

Figure 2.6: **A quadrupedal wall climbing agent.** Evolved with the SAGA algorithm and a simple neural network, the Stanton *et al* quadrupedal agents had to climb over obstacle walls of incrementally varied height to reach their goal [80].

each section of the image, "sliding" them across. They evolved the design of these grids using the network, and patterns emerged on them that were capable of detecting particular features. The pooling layers then measured which features were in which area and passed this on to the next convolutional layer. This process produced complex multi-faceted features across multiple layers. Finally, the last feature map was passed on to the decision network, which analysed which features were present in which areas of the final processed image, matching it to its most likely class. In order to prevent the network producing filters relying on too many neurons, neurons were randomly dropped out with a probability of 0.5. They chose this architecture as it functioned well with prior knowledge, to accommodate the vast training sets used for the image database. Parallel processed through a pair of Nvidia GTX GPUs, the network could be trained in 5-6 days using images scaled to 256x256 pixels. The network was even able to match images to the same class that were semantically familiar but distinctive in nature. Furthermore, when they removed the depth of the network, reducing the number of convolutional layers, the results began to suffer in quality.

Following this, Mnih *et al* produced the first deep learning model to learn from

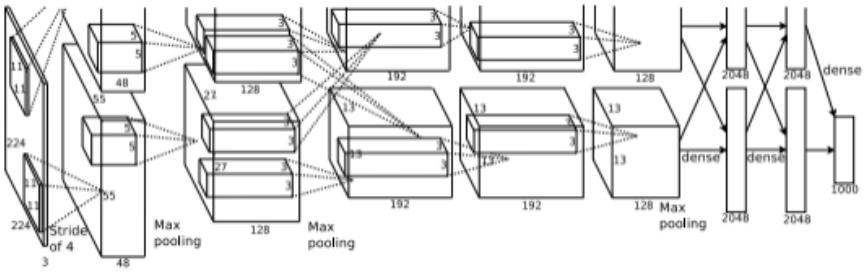

Figure 2.7: **The deep architecture used by Krizhevsky *et al.*** The input image (far-left) was processed through 5 convolutional/pooling layers (cuboids) using kernels (inner squares/cuboids). At each layer, progressively more complex kernel features were learnt and pooled together (cuboid depth), before being processed through a 3-layer fully connected decision network (final rectangles), classifying it as one of a thousand image types [42]. The network is split into two rows, one for each GPU used, with several connections to synchronise them.

high dimensional input via reinforcement learning (instead of evolution) - Deep Q learning [55]. They adapted an existing reinforcement learning algorithm, Q-learning, to learn a neural network mapping for a set of non-similar Atari games, using the screen image as input, and the gamepad buttons as output. Q-learning uses the learnt-over-time cumulative reward value of probable states reached from each action, labelling this the quality or "Q" value. This allowed the network to be updated to choose the ideal Q-value for each given situation. Using a similar deep network to Krizhevsky *et al* together with Q-learning, they updated their weights using stochastic gradient descent, and used random experience replay to help the CNN with sparse rewards. The score was normalised across all the games, with frameskip enabled to assist runtime. They achieved good scores on most games and beat experts on three (breakout, enduro and pong). These games were the ones with the shortest timescales, requiring the least planning.

There have been further examples of deep learning producing successful virtual bipedal agents and performing well on other continuous tasks. Lillicrap *et al* adapted Deep Q learning to a continuous domain via an actor-critic policy gradient algorithm [46]. This algorithm used two separate learned functions for reward estimation and parameter updating. It solved 20 simulated physics tasks including legged locomotion and complex object manipulation. Jolley *et al* studied the use of convolutional neural networks on the aforementioned river crossing task [40]. They produced a 99% completion rate on the most difficult iteration of the task and even outperformed

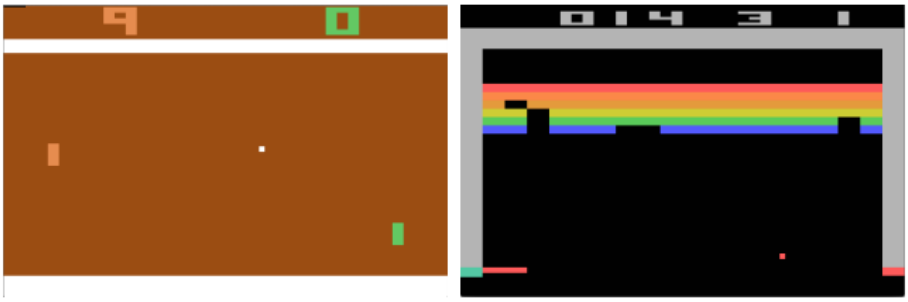

Figure 2.8: **Short-term planning games.** Pong(left) and Breakout(right) both produced higher scores than human experts as they required little long-term planning [55].

the previous handcrafted solution. These further demonstrate the potential of the methodology for the bipedal task, and set the bar for agent-based behavioural simulations.

Another important aspect of agent-based systems, perhaps one that isn't immediately obvious, is the choice of physics engine for the simulation. There are many aspects to a physics engine, and each is often designed with a different one in mind. Engines aimed towards games development are usually more focused on material properties and multiple simultaneous bodies, whereas simulation engines are more concerned with numerical accuracy and joint constraint enforcement. In 2007, Boeing *et al* investigated accuracy and efficiency in a number of physics engines, including Novodex, Bullet, JgLib, Newton, ODE, Tokamak and True Axis [7]. Testing the integrator functions (functions estimating the next timestep from a list of bodies and forces), material properties, stacks, joints and collision detection, they concluded that no one physics engine was the best at any test, and the best engine to choose would depend on your task. Interestingly, every engine failed the sphere stacking test, as none of them included noise, leaving the spheres perfectly balanced. More recently in 2015, Erez *et al*, creators of the Mujoco engine (although they primarily use objective measurements to avoid bias) compared Bullet, Havok, Mujoco, ODE and PhysX for model-based robotics [29]. They tested integrator functions at low time-steps to observe derivative error, and also tested a number of tasks emphasising different aspects of engine quality. Seen in Figure 2.9, these included a grasping hand, a large number of loose capsules falling to the floor, a long chain of links and a humanoid body moving at regular intervals. They discovered that Mujoco was the fastest and most stable, but lost its advantage in contact and collision dynamics, further emphasising

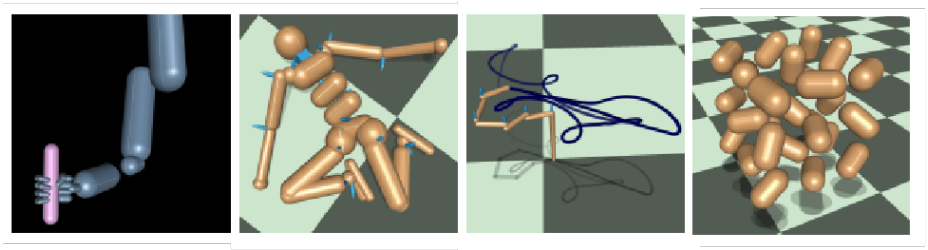

Figure 2.9: **The robotics tests employed for various modern engines.** Four tests utilised by Erez *et al* for different aspects of a physics engine. (left to right) The grasping task, a regularly moving humanoid, a series of linked bodies, and dropping several small capsules at once.

the point that engines must be carefully chosen with purpose in mind. In the context of agent-based systems, this could also mean that moving from one engine to another may produce different results for the same task, and suggests that engine choice must be carefully taken into account when investigating behaviour.

2.3 Summary

This Chapter addresses the history of the relationship between genetic algorithms and neural networks, citing notable works featuring the two together. I discuss the most infamous agent based evolved neural network controller system, Sims' virtual creatures. Sims' later work featuring creatures fighting for position is also discussed alongside co-evolution. This demonstrates the idea that behaviour can be evolved through neuroevolution. I investigate the importance of embodiment within agent based systems, covering the Callisti *et al* study into octopus morphology and the Bongard *et al* investigation into object manipulation. This demonstrates the benefits of morphology for bipedal locomotion. With bipedal walking involving a notoriously tough fitness landscape - I aim to utilise morphological adjustments to ease computational burden on my neural network controllers. Incremental evolution is also demonstrated as an aspect of evolutionary systems - I aim to display this as a perk of evolutionary processes. Finally I demonstrate deep learning systems, the bar they set for the field of neural networks, as well as demonstrating the impact of physics engine choice on a task. Choosing my engine carefully - I aim to move toward the level of complex behaviour seen from these modern systems, and produce bipeds evolved with a neural network that go beyond simple forward locomotion.

Chapter 3

Background - Bipedal Walking

3.1 Bipedalism and How It Evolved in Us

Bipedal Walking, sometimes referred to as bipedalism, is defined as the locomotion of species that stand on two legs. Darwin's Origin of Species demonstrated that the human race descended from apes, creatures that walked on four legs [24]. Therefore at some point we must have moved away from quadrupedalism. The primary theories for our transition are to free up energy for endurance running [9] and to free up the hands for tool usage [36]. Unlike quadrupedalism, bipedalism requires much more balance; fewer points of contact with the ground means the centre of mass has much less room to move around without causing toppling. In nature, a bipedal gait is usually balanced out with a swinging tail or passive arm swinging [2]. Many works of research have recreated our gait, both through robotic and simulated agents. Van de Panne *et al* guided virtual bipeds with varying numbers of actuators through walking gaits via parameter optimisation and a harnessing vector dubbed the "hand of God" [84]. Kanehiro *et al* created a two-armed bipedal robot capable of walking and carrying objects [41]. Behnke *et al* generated omni-directional trajectories for online bipedal robots [4].

The classic example of a bipedal robot is Honda's ASIMO [67]. Since 1986, Honda have been building humanoid walking robots; by 2000, they had refined their robots into the first iteration of ASIMO, with an upper torso and an astronaut-like design. Honda finalised the latest iteration of the machine in 2010, featuring further balancing and emotive capabilities, and most recently, a basic intelligence based around its active vision system. The most notable modern bipedal robot, however, is likely the Atlas robot, from Boston Dynamics. Performing complex acrobatic motions, it can

remain stable throughout various routines [27]. It was also created as a research platform, designed to be programmed and simulated worldwide. It features 3D printed components designed to be lightweight and compact. Boston Dynamics produced several methodologies for optimising its control via machine learning, pairing it with modern intelligent systems.

In silica, Azarbadegan *et al* looked to test some of the hypotheses for how bipedalism evolved in the natural world by evolving bipedal agents in a Sims-like system [2]. To assist in the emergence of bipedalism, they encouraged falling in their system, as most bipedal gaits involve falling from one foot to the other. To prevent static agents from falling without walking anywhere, they introduced a damage penalty to the system. To further encourage bipedalism, the two limbs with the most amount of damage were exempt from the penalty. Creatures were required to have a head component as well as limbs. They reduced fitness for how far the head sunk from its original height. Most of their results were "cheaters", growing taller to increase height and then falling further than the others. To combat this, they introduced a minimum fitness to be included in the initial generation, in an attempt to produce genetic diversity in subsequent generations through a more even fitness distribution. They produced two successful bipeds. The galloper was a triangular creature that hopped with its back foot, whilst landing on the front foot. The crawl-stepper balanced on one cuboid limb and rotated its hips to crawl along on each corner of the balancing limb. Whilst not technically bipedal, they described the behaviour as similar due to its nature, falling onto the balancing limb over and over. Despite their bipedal agents evolving successful gaits in a Sims-like system, their morphologies were still far from traditional symmetrical bipedal bodies.

As well as this, Geijtenbeek *et al* demonstrated a method of controlling bipedal agents using simulated muscle fibres [31]. They optimised the location, strength and activation times of simulated muscles around a set of varying model bipedal skeletons using a genetic algorithm. They modelled the fibres in each leg as a simple finite state machine. To transition between states and walk, they generated a set of target poses based on the walking gait of the current template, and then computed the required muscle excitations to move through them. They were able to produce muscle structures supporting both walking, turning and running gaits on both human-like and ostrich-like models. These structures also proved to be robust to external forces, and on steep slopes. Despite achieving robust gaits in multiple bipedal morphologies,

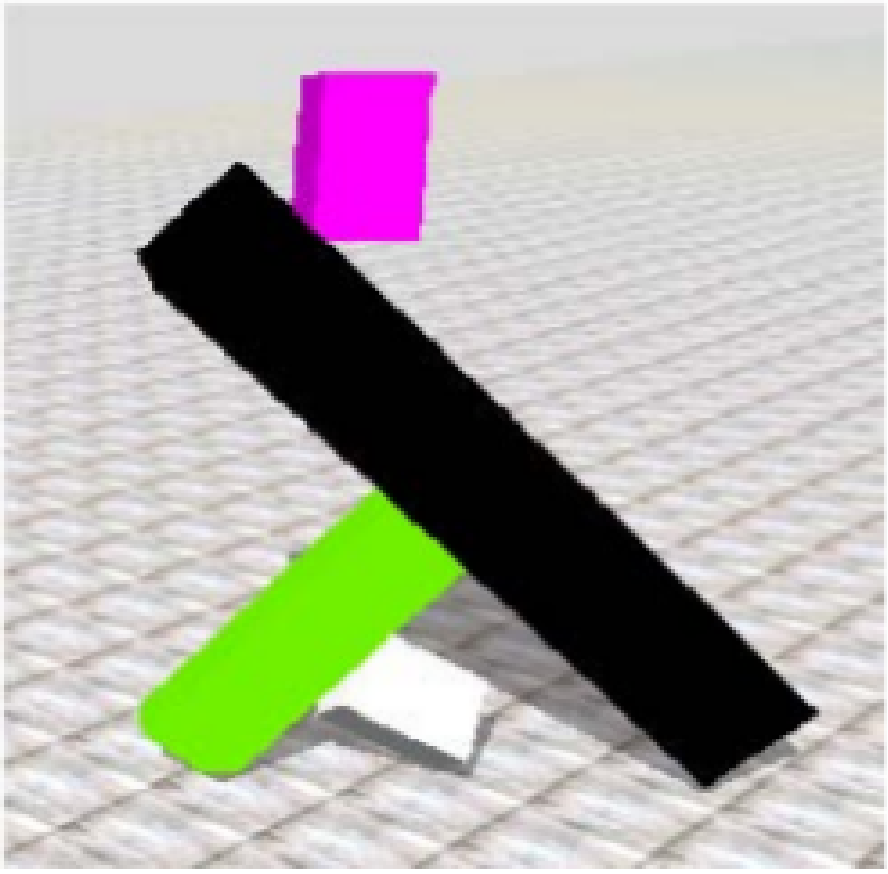

Figure 3.1: **A high performing biped** produced by the Azarbadegan *et al* Sims-like system [2]. Evolved in a similar way to Sims' creatures, it gallops forward with its back green foot, landing on its front black foot. Its head is shown in purple, kept high up to avoid a reduction in fitness.

none of their agents utilised any form of neural network control beyond than the finite state machine model.

Lehman *et al* investigated novelty search for bipedal walking as one of two experiments to further understand the relationship between evolution and complexity [43]. They argued that the relationship between evolutionary fitness and complexity was not linear, with complexity not always favoured by selection, citing Miconi *et al* [52] and Nowak *et al* [59] as examples. They believed complexity to instead be an inevitable product of random selection. They chose new individuals based on the sparseness of their location in a novelty space filled with a selected archive of high novelty individuals. They also employed NEAT [78] to select their agents neural network controllers. They utilised two main experiments to compare fitness and novelty search. The first of these was a deceptive maze task. Deceptive tasks are defined as evolutionary tasks with local fitness minima designed to trap a genetic algorithm and keep it from the global fitness peak. The first experiment featured two maps. The first map contained lots of dead ends near the goal, whilst the second required a path that travelled far from the goal. They measured fitness as distance to the goal, and novelty as the difference in neural control networks. Novelty search was able to find a solution that was three times faster and much less complicated. However, when they unenclosed the map, removing a portion of the outer walls, novelty search failed as it saw everywhere outside the maze as a new position. The second experiment featured novelty search for bipedal walking. Their bipedal walker modelled had 6 actuated joints and a recurrent neural control network. They measured fitness as distance travelled in any direction, and the novelty metric as the difference in neural control networks. Novelty search produced more oscillatory agents able to travel a much further distance. They did not investigate behaviour beyond forward walking.

Morphology also plays a role in bipedal locomotion. McGeer designed a walker which travelled down a slope using gravity as its sole source of energy, moving its legs in a natural swinging motion [51]. Dubbing this a "passive dynamic walker", it highlighted the potential of replacing sections of a control process with morphology. This passive leg swinging was also very similar to the way humans swung their legs below the knee as they walked. Paul *et al* investigated optimising both the the neural network controller and morphology of virtual 3D bipeds at the same time [62]. They controlled their bipeds using recurrent neural networks, featuring proprioceptive sensor inputs for each joint and an additional haptic sensor input for each foot. Each

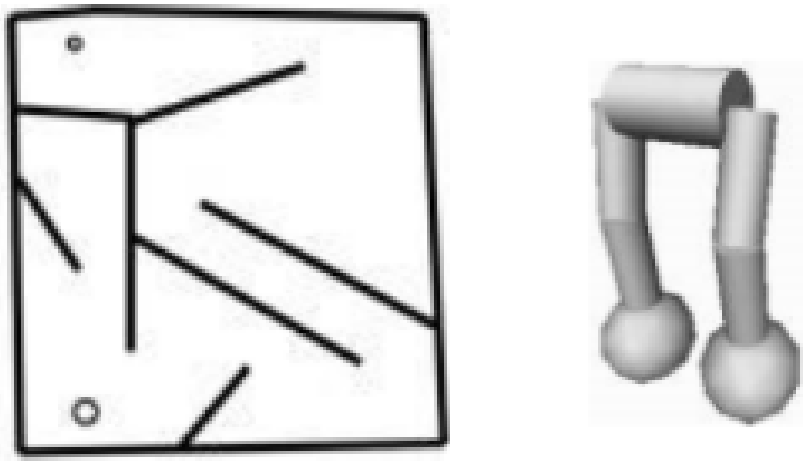

Figure 3.2: **The two tasks Lehman *et al.* used to test novelty search.** (left) The deceptive maze task forced agents to move away from the goal in order to eventually reach it, making it difficult for standard evolutionary processes. (right) The bipedal agent used in the locomotion task. It aimed to travel as far from its origin as possible without falling [43].

biped had a certain portion of mass that could be distributed through various blocks on its body. These blocks had length, width, location and individual mass parameters. Symmetry was enforced to ease task difficulty. They modelled their genotypes as a string of floating point values encoding both the network weights and the biped's mass block parameters. They discovered that across their tests, the biped populations with the largest amount of distributed mass featured the highest proportions of bipeds capable of stable locomotion. No further behaviour was investigated beyond forward locomotion.

Collins *et al* investigated adding dynamic arm swinging to a bipedal walker in order to further mimic the human gait [19]. Their walker had swinging arms attached at the hip and an inter-leg hip spring. Searching for limit cycles that would allow for stable locomotion, they were able to find gaits with phase, anti-phase and parallel arm swinging. These gaits all exhibited a reduction in vertical angular momentum from arm-less gaits. They went on to bind human subjects' arms to test the effects these gaits had on their metabolic energy consumption. They tested arm swinging in phase, out of phase, bound to waist, and held to waist. They discovered that all of these besides in phase caused greater vertical angular momentum and significant increases in metabolic rate.

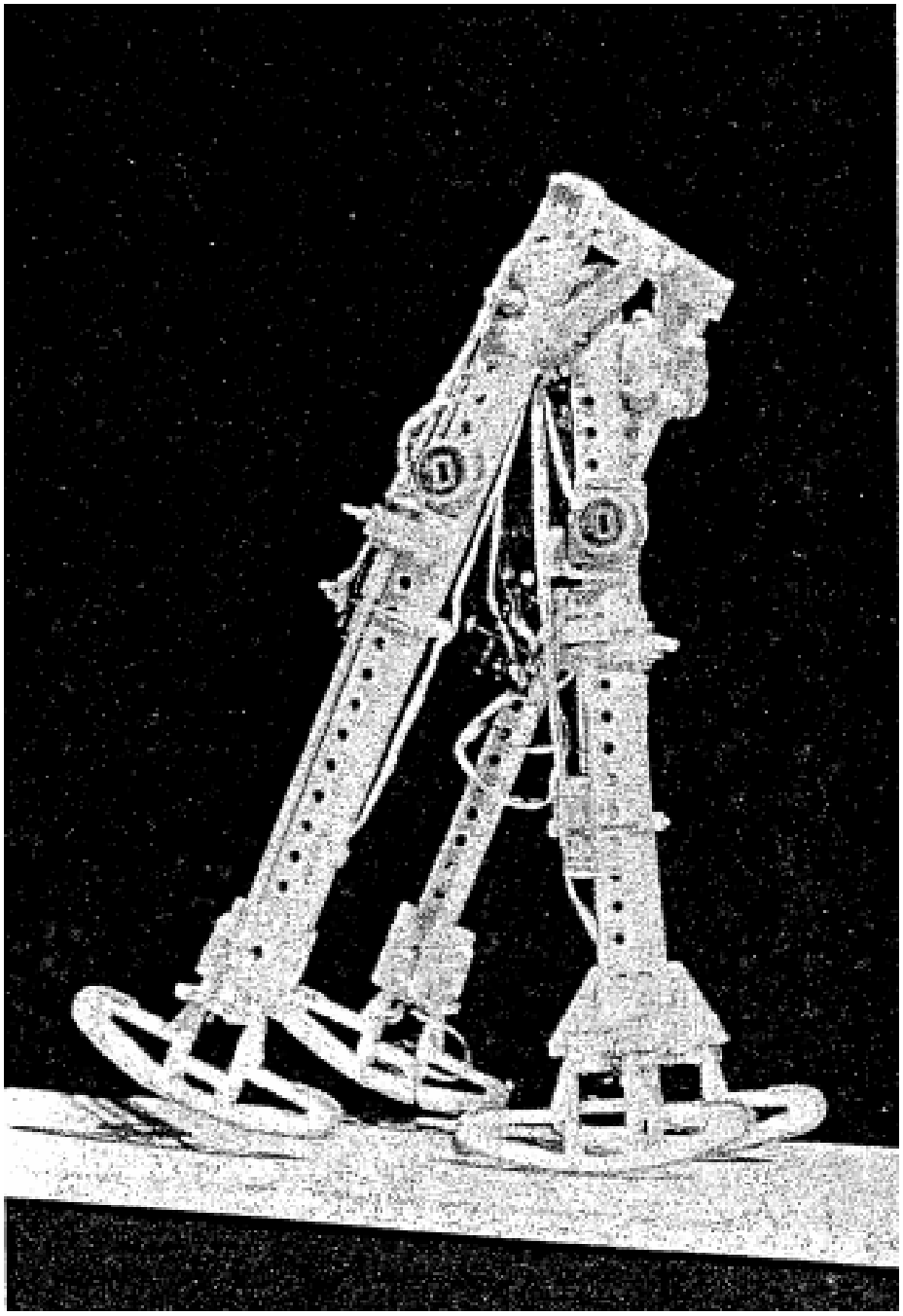

Figure 3.3: **The original passive dynamic walker designed by McGeer.** It was capable of walking motion down slopes through gravity alone [51].

3.2 Neural Bipeds

One of the most common neural network controllers for evolving bipedalism is the Central Pattern Generator. Central Pattern Generators are networks that produce rhythmic output from a linear input. Delcomyn described how real pattern generators, often located in the spine, which CPGs are based on, aid the oscillation of walking creatures via an electrical pulsation from the brain [25]. Reil *et al* used a CPG network to evolve bipedal walking [64]. They controlled their walkers with ten interconnected neurons, each with a modulating time constant. The walkers were evolved using paired tournament selection. The genotype for the system was modelled as the network weights and measured fitness as total distance from the point of origin. Success was defined as stable walking for 50 seconds. After 100 runs of 100 generations, they achieved a 10% success rate amongst the evolved walkers. High levels of diversity were produced amongst the successful gaits. Some travelled backwards instead of forwards, and many were asymmetric. They then decided to apply a fitness bonus for oscillatory movements. This brought success rates up to 80%. Seeking to improve robustness further, they added sensors to the hips of the walkers and redefined their fitness as distance to signal source. This produced walkers capable of turning their hips toward the source before walking to it. Unfortunately, the walkers gaits became unstable close to the source. No further behaviour was investigated beyond forward walking and singular lateral turning. They also did not utilise morphology to simplify their evolutionary process.

Wolff *et al* evolved a network of CPGs to achieve robust bipedal gaits in 3D, with temporary support [93]. They set their CPGs period to that of a normal human walking gait, and doubled the frequency for the knee and waist joints. They then evolved their weight values with a genetic algorithm, modelling fitness as the distance travelled forward minus the distance travelled to the side. Their walkers featured both proprioceptive and foot contact sensing. Starting with a four point support frame, the walkers evolved to rely on the support, and all runs failed when it was removed. With a two point frame, they were able to achieve indefinite walking, although hip and ankle joints had to be frozen for additional stability. They also attempted to remove the two point support after the first second of simulation. This achieved some fast gaits, but most were unstable. They did not investigate any further behaviour beyond forward walking.

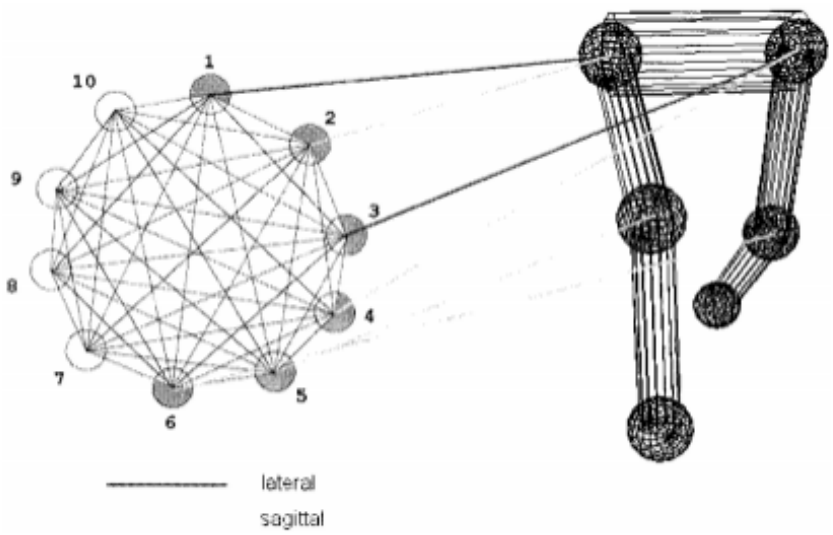

Figure 3.4: **The CPG network and bipedal agent designed by Reil *et al.*** (left) The interconnected network of ten pulsing CPGs, of which four provide lateral and sagittal output to the hip joints, and two provide sagittal output to the knee joints. They evolved the weights of this network to make their biped model (right) walk [64].

Vaughan *et al* utilised both passive dynamics and Central Pattern Generators to produce agents with robust bipedal gaits in a 2D physics engine with adjustable speed [85]. Noting that human bipedal motion is both dynamic and analogue, they constructed a simulated biped modelled after a pair of human legs, with flat feet, two gyroscopes and a CPG. They assigned each section of the leg a network, not connecting them to narrow the behavioural search space. One leg was given inverse inputs to encourage symmetrical behaviours. They used a genetic algorithm, modelling fitness as a function of distance penalised for excessive derivation from the initial heading. They then tested the walker to see if the combination of the CPG and evolved parameters had produced a gait capable of passive locomotion. They placed the highest performing walker on a slight incline, and turned on the CPG. Allowing the machine to take a single step down the slope, they then disconnected the neural control network. This was successful: the walker was able to walk down the remainder of the slope passively.

Following this, Vaughan *et al* investigated producing a gait with the 2D system that would be sensitive to the CPG powering it. They first re-evolved the walker without sensor feedback to make sure the system was as sensitive to the CPG as possible. They then pulsed the CPG using the foot strike timing of the previous

passive walking experiment, and placed the machine on a slight incline. They turned off the sensors until the machine had taken ten steps to encourage the passive part of the gait. Finally, after reconnecting the sensors, they slowly reduced the incline over generations until the machine was walking on a flat surface. The walker that emerged from this process was a stable machine that did not fail even after thousands of steps. they noted that the lower half of the evolved legs swung with gravity, much like real bipedal organisms. The gait was robust to both external and internal noise. They could adjust the pulse of the CPG and the walker would adjust its gait to match, demonstrating its evolved sensitivity.

Vaughan *et al* attempted to improve their biped further in order to walk in 3D and on rough terrain. They added ankle joints to accommodate lateral motion. They extended the previous model to 4 separate networks in each leg for each of four walking states: contract, swing, extend and support. Switching neurons connected the networks; the most active switching neuron in each leg represented the state the leg was in. The two combined leg network sections remained unconnected. They initialised the walkers with a contract leg and a support leg until one of the feet left the ground. Testing them first in 2D, their passive walkers produced high fitness just from pulsing the CPGs without evolution. The fitness on rough terrain was much lower, although successful walkers were still produced. They also encountered a problem in which the walker lost momentum in both legs and toppled over. In order to address this issue, they created a fifth "stand" state to contract the non-supporting knee if the machine stopped both legs. This then allowed them to start the walker from a standing upright position without having to initialise the legs as contract and support. They then tested the walkers in 3D. They lowered the mutation rate of the genetic algorithm to facilitate the more complex fitness landscape. The walkers still exhibited high fitness on a flat surface and lower fitness on an irregular one. They noticed that the walkers' feet were now capable of moving in front of each other, tripping them up. To fix this, they implemented a foot tangle network connected to both legs, that inhibited a neuron if it produced a torque value that would result in one leg moving in front of the other when applied. This was determined by walker leg angle and velocity data. This was somewhat successful, increasing the rough terrain fitness slightly. Vaughan added a full upper body to balance atop the walker and test it further. Even with a head, torso and passive swinging arms the walker was able to evolve a fit enough gait to walk along a flat surface indefinitely. They did not investigate any further behaviour beyond forward walking.

Wiklendt *et al* used a spiking neural network to control 3D bipedal walkers [91]. They used a leaky integrator function in each neuron, which featured a potential firing value that affected output. This value lowered over time but increased with signals from nearby neurons. They evolved the networks using a mu-lambda strategy and gave the walkers an initial nudge of force. Their walkers' knees swung automatically. Of the five runs performed, three showed stable gaits, reaching a maximum distance of 16 metres. By measuring the timings of neuron spikes and pigeon-holing them into intervals, they were able to demonstrate the emergence of periodic spiking in the networks. They generated neural control networks for humanoid characters, using only their physical parameters and a fitness function [1]. They modelled their genotypes as weighted directed graphs, much like Sims. These were then translated directly into neural control networks (node to neuron, edge to weight) and evolved with NEAT. They produced neural network controllers capable of smooth motion for a number of different humanoid morphologies, although none of them were able to travel far without toppling.

In order to evolve walking in 3D on rough terrain, Solomon *et al* assigned a linear reactive controller neural network, seen in 3.5, to each desired angle. Their walkers consisted of 9 sections - two hips, two thighs, two calves, two feet and an upper body. These were linked together with 8 joints, of which, the hip, knee and ankle joint pairs were actuated. Desired angles were defined as angles in the walker model that were chosen to be adjusted over time in order to produce motion. Seen in 3.6, their chosen desired angles in the 2D model were the upper body, inter-leg, stance knee, swing knee, stance ankle and swing ankle angles, highlighted in red. In 3D, an additional desired angle was added: lateral inter-leg angle. Also seen in 3.6, the stance leg was the leg contacting the ground, whilst the swing leg moved forward. These roles then alternated as steps were taken. Each neural network controller was fed the inputs seen on the left of 3.5, consisting of normalised model angles, normalised model angle differentials (velocities calculated from the previous time-step), a boolean value representing the condition of both feet being in contact with the floor (double-stance), a network mutation rate, and a bias term. Normalization in this context was applied to the model angle inputs and model angle differential inputs, transforming them between the range of -1 and 1. All the inputs were then multiplied by a set of weights which were evolved using a genetic algorithm and then summed to produce a desired

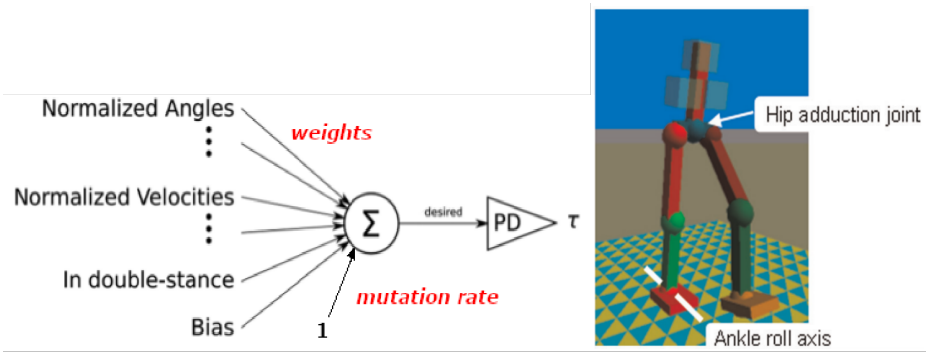

Figure 3.5: **The network topology and biped designed by Solomon *et al.*** (left) Their networks featured a simple summation of inputs, multiplied by their respective evolved weights, alongside an evolved mutation rate, to produce a torque value for each desired angle. (right) The model produced by them can be seen with labelled lateral joints walking on rough terrain [77]. It also featured six sagittal joints at the hips, knees and ankles.

value. This value was then passed through a PD control, which applied error correction to the desired value proportional to the previous error and the differential of the previous error (P and D), in order to produce a final torque value which was then applied to each of the joints of the model necessary to adjust the desired angle. For the upper body and inter-leg desired angles this torque was split evenly across both hip joints, whilst the remaining desired angles (stance knee, swing knee, stance ankle and swing ankle) only required the torque to be applied to a single joint to adjust the angle. The 2D joints in the model were situated at the hips, knees and ankles, with the 3D joints adding lateral actuation to the hips and passive roll to the ankles.

Solomon *et al* tested four neural control network variations: Local Proportion, where desired angle specific inputs were connected to each actuator, Fully Connected, Sparsely Interconnected (SI), where the Fully Connected network was pruned until fitness declined, and finally Reduced Sparsely Interconnected (RSI), where the inputs were set to those that were prevalent in at least 80% of pruned networks. Fitness was measured as distance travelled forwards in the X dimension. Walkers were terminated when they either fell over (a part of their body above the knee contacted the ground) or exceeded a given amount of torque across their lifetime, precisely 3000 units. They modelled their genotype as the network weights across every neural network controller, initialised between 0.05 and -0.05. Every generation, the genetic algorithm simulated the population and sorted it by fitness, replacing the population (save the top performing elite individual in 2D simulations) with offspring produced through

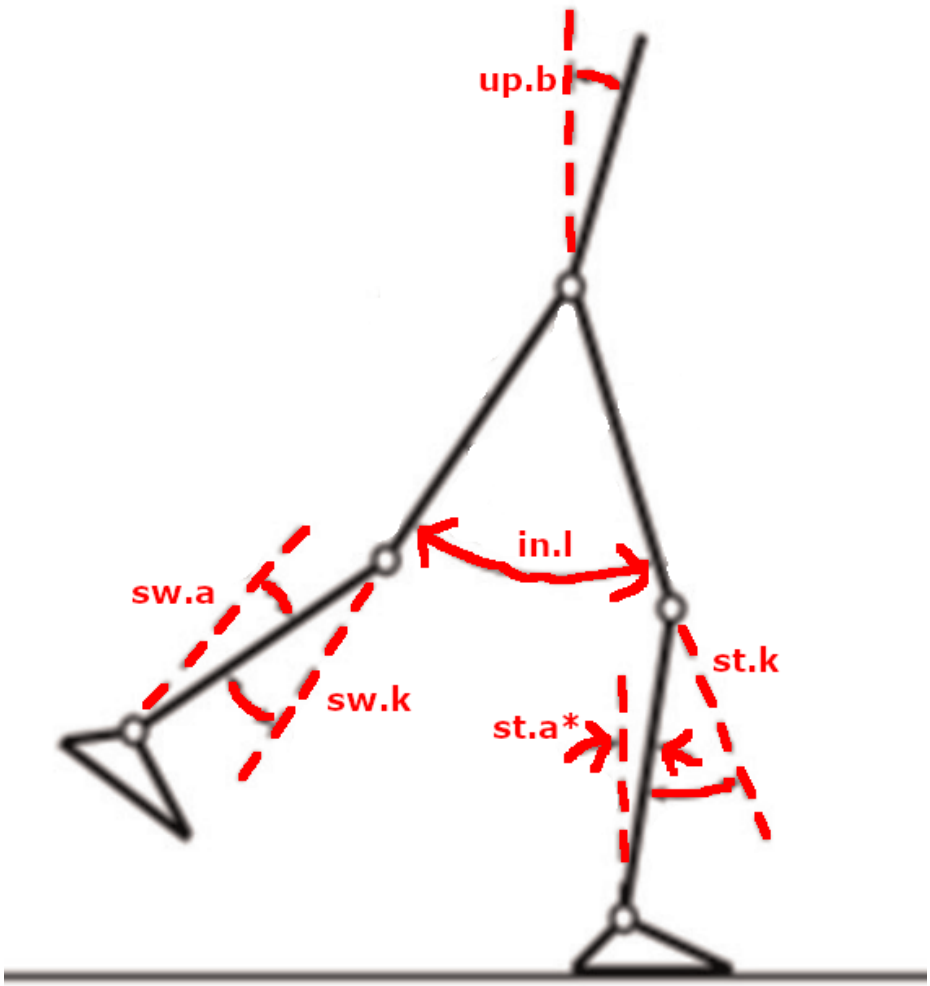

Figure 3.6: **Desired angles in 2D.** Solomon *et al* chose six desired angles for their 2D model. By applying torques to the relevant joints, they adjusted these angles to produce motion. Seen in red, the 2D angles chosen were the upper body, inter-leg, stance knee, stance ankle, swing knee and swing ankle angles. Note that the stance ankle angle was inverted. In 3D, an additional lateral inter-leg desired angle was added.

either mutation from the top 20%, random crossover from the top 20% or both. The mutation rates for each neural network controller were also evolved over the course of the simulation as an additional part of the genotype. The mutation rates were initially sampled from a normal distribution with a random mean between 0.05 and -0.05. Mutations were applied as the mutation rate multiplied by another random normally distributed value; the result of which was then added to the genotype. As rough terrain was generated randomly, fitness was not consistent, so they chose the winning walker from the top 20% of the final 10% of generations. They bootstrapped the Fully Connected walkers with successful Local Proportion neural network controllers from a previous 2D physics engine simulation incrementally. From there, they were able to prune down to the RSIL neural network controllers and produce successful 3D agents on rough terrain. In 3D they added hip adduction and passive ankle joints, as well as an additional neural network controller for the hip spread desired angle. They achieved smooth 3D walking on uneven terrain with a maximum height of 2% leg length.

Manoonpong *et al* built a bipedal robot that utilised powered passive dynamics and a neural control network to adapt to different terrains [50]. Santos *et al* utilised motor primitives, or sub-movements, that could be combined with an arrangement of optimised CPGs to produce quadrupedal locomotion [70]. They were able to produce walking, turning and following behaviours. Santos *et al* then tested centre-crossing CPGs for bipedal walking [71]. They centered every neuron's activation function over the range of its inputs. They then set the network biases to the negative sum of all the weights, making the network maximally sensitive to change. This resulted in a powered biped with a stable gait. The biped was also stable in slight noise, although they had to tune the biases carefully. Silva *et al* generated 2D bipedal locomotion neural network controllers based on CPGs [73]. They used genetic programming on tree-like neural network controllers to form a set of motion primitives, similarly to Santos *et al*. These controllers achieved stable gaits able to adapt to curving slopes. Missura *et al* utilised an on-line learning strategy based on error feedback from desired motion, together with a CPG, to enable a bipedal robot to learn its balance in a few steps [54].

In recent years, deep learning has been applied to bipedal walking. Arguably the benchmark for bipedal behaviour, Heess *et al* used their Distributed Proximal Policy Optimisation reinforcement learning algorithm to produce bipedal agents capable of performing complex tasks [35]. Distributed Proximal Policy Optimisation was based

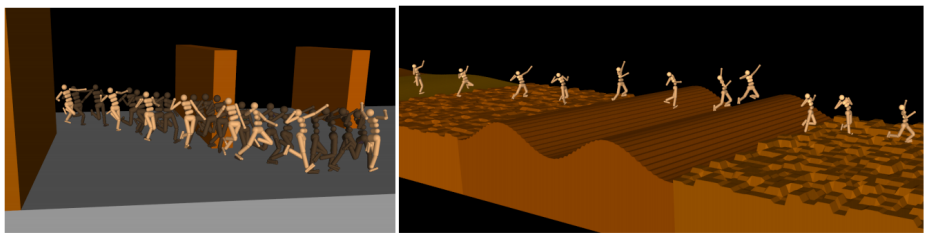

Figure 3.7: **The running and jumping behaviours produced by Heess *et al.*** (left) Their humanoids slalomed around walls. (right) Their humanoids ran and jumped over rough terrain and gaps [35].

on a different policy gradient algorithm, Trust Region Policy Optimisation, modified for recurrent neural control networks and vast parallel processing. Most policy gradient algorithms operate by maximising an expected sum of rewards with respect to the parameters of a control policy. The "Trust Regions" then represented a limited amount of change (the trust to not move too far) in policy parameters per update. They believed task-specific rewards could be victim to over-fitting and did not encourage learning. They thus used a simple reward signal only based around moving forwards, complexifying the environments the walkers were situated in instead of the control process. Their agents had two separate neural control networks, one for their local sensors, such as joint angles, and the other for their environmental sensors, such as nearby terrain height. They were able to produce running, jumping and even slalom behaviours (moving left and right around obstacles whilst moving forwards) as the walkers travelled forwards. They also produced quadrupedal and planar walkers capable of navigating obstacles and jumping over gaps.

Salimans *et al* used an Evolutionary Strategy as an alternative to Distributed Proximal Policy Optimisation for bipedal walking [68]. They used a natural evolution strategy, modelling population as sets of network parameters produced with hyper-parameters. Each generation, the strategy evaluated a number of random perturbations of the parameters produced by the hyper-parameters. The results were then combined, the gradient estimated and the hyper-parameters updated for the next generation. To allow the algorithm to run evaluations across parallel units, they synchronised the noise for each member of the population and transferred between master and worker threads. They were able to match Trust Region Policy Optimisation for certain tasks, including humanoid walking, at a greater runtime. They stated that this demonstrated the ability of simpler classic methods scaled up with modern hardware. Their evolutionary strategy also produced more complex solutions for the

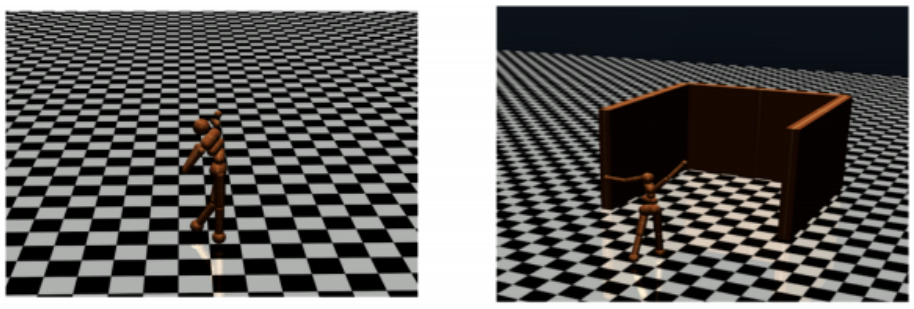

Figure 3.8: **The humanoid environments used in Salimans *et al* and Such *et al* .** (left) The basic 3D humanoid walking environment in the Mujoco engine. Agents were only required to travel forwards without falling. (right) The hard maze environment solved by the weighted novelty search used by Such *et al*. In order to travel forwards, agents must temporarily abandon fitness in order to go around the trap. This was difficult for standard evolutionary processes.

simpler 3D tasks, and performed better in almost half of the Atari games. Despite their success at the humanoid walking task however, the walkers did not exhibit life-like gaits, instead displaying shuffling behaviours with minimal motion. Conti *et al* added to this work by using novelty search together with the deep evolution strategy and explored different weightings of novelty against fitness [20]. They achieved even higher Atari and humanoid performance.

Subsequently, Such *et al* used a genetic algorithm on the same problem (3D Humanoid-v1), using two 256-unit hidden layers (matching the configuration file included in the source code released by Salimans *et al*), achieving success on this task but noting that their GA "took 15 times longer to perform slightly worse than ES" [82]. They also noted that (while only just qualifying as a deep neural network, having more than one hidden layer) this network contains approximately 167k parameters, orders of magnitude greater than the previous largest neural networks evolved for robotics tasks. They encoded these parameters using a novel method that stores, for each genotype, an initialization seed and a list (that grows with each mutation) of random seeds used to generate mutations to the vector of parameters. They also applied a weighted novelty-search version of the ES, which was able to produce high scores in a deceptive environment that required the humanoid to walk around a trap, demonstrating that ES is more robust to parameter perturbation in the humanoid locomotion task than both their GA and Trust Region Policy Optimization [20].

3.3 Summary

This Chapter addresses historical bipedal systems and common methodologies used with bipedal agents. Initially I discuss theories for why bipedalism emerged in humans - to allow more energy for endurance running, and to free the hands up for tool use. Next I discuss notable bipedal robots ASIMO and Atlas, and some examples of simulated bipedal systems from Azarbadegan *et al* and Geijtenbeek *et al*. I then cover the successes of novelty search and passive dynamics in bipedal systems. Finally I cover evolved neural network controller bipedal systems: the CPG-based walker from Reil *et al*, the Linear Reactive controller neural network based walker from Solomon *et al*, and most recently, the Salimans *et al* deep evolutionary bipedal walker. Each of these systems are highly successful in their own right, but usually are either not utilising their morphology to aid their gait, lack a neural control network, or lack evolutionary process. Very few utilise all three techniques to enhance their bipedal gaits. As such, I aim to enhance existing bipedal systems, and produce successful bipedal agents with neural network controllers using evolutionary techniques. My bipeds will also have symmetrical, human-like bodies, and utilise additional morphology in a similar fashion to passive dynamic systems. The other caveat in these bipedal systems is that most lack complex behaviour beyond forward walking. If this could be achieved, evolutionary systems could produce the behaviours seen from machine learning with the additional perks only seen in evolutionary systems. As such, I also aim to evolve robust, life-like gaits and complex bipedal behaviours without any prior bootstrapping.

Chapter 4

Replacing Prior Bipedal Bootstrapping with Reduced Control Penalty and Elitism

4.1 Chapter Aims

- In this Chapter I aim to replace the 2D bootstrapping required in the Solomon *et al* Linear Reactive system with fitness function enhancements, to produce an enhanced system simple enough that it can be used as a straight-to-3D base for the future bipedal walking works of myself and others in the academic community [77]. This is in line with my aim to increase walker fitness.
- I also aim to produce behaviour in the system beginning to look beyond forward walking. This is in line with my aim to produce complex behaviour.

4.2 Methodology

4.2.1 Rationale

In order to produce a starting point for myself to produce robust and complex behaviour in bipedal agents, I initially chose to re-implement an existing system. I chose this as starting from a successful system would allow for more time to experiment with potential methodologies without having to spend time on configuring results that already exist. An alternative would have been to build my own system entirely from scratch, but this would be more complex and also less relevant than starting from an established point. I then chose to re-implement the Solomon system. This was chosen as the Solomon system had achieved robust results via rough terrain using an extremely simple neural network controller type. This simple structure not

also further demonstrated robustness of behaviour, but also left room open for future modifications. An alternative system could have been the Reil system seen in [64], but this would have used a more complex neural network controller; an arrangement of ten fully connected pattern generators. The main issue with the Solomon system was the prior bootstrapping required. Solomon *et al* used pre-evolved genotypes from a 2D system to initialise the 3D walker. This meant that a large amount of pre-processing was required. In order to use this system as a starting point that could be easily used by future researchers, I produced a version of this system that could be evolved from scratch in 3D. To achieve this, I chose to use fitness function modifications to improve fitness enough that initially high-fitness genotypes were not required. I chose to use enhancements modifying fitness functions and task rules as these can be seen as the most direct way to influence an evolutionary system. This can be seen from Sims, where fitness is changed in the famous Virtual Creatures system to the control of a cube, fundamentally changing behaviour [74]. This can also be seen from Chaumont *et al*, who modified the fitness function in the Sims system in order to produce catapult behaviour [16]. They also introduced coefficients based around agent size and complexity to the function of the original system to generate a more diverse population to sample. An alternative would have been to use morphology or passive dynamics as stated in my other main aim, but this would have been a complex change to produce a simple starting point.

4.2.2 Re-implementation

In order to match the original work of Solomon *et al*, my system was coded in the ODE 3D physics engine [76] [77]. To evolve walking in 3D on rough terrain, Solomon *et al* assigned a linear reactive controller neural network, seen in 4.1, to each desired angle. Desired angles were defined as angles in the walker model that were chosen to be adjusted over time in order to produce motion. Seen in 4.2, their chosen desired angles in the 2D model were the upper body, inter-leg, stance knee, swing knee, stance ankle and swing ankle angles, highlighted in red. In 3D, an additional desired angle was added: lateral inter-leg angle. Also seen in 4.2, the stance leg was the leg contacting the ground, whilst the swing leg moved forward. These roles then alternated as steps were taken. As in the original, each neural network controller was fed the inputs seen on the left of 4.1, consisting of normalised model angles, normalised model angle differentials (velocities calculated from the previous time-step), a boolean value representing the condition of both feet being in contact with the floor (double-stance), a network mutation rate, and a bias term. Normalization in this context was

applied to the model angle inputs and model angle differential inputs, transforming them between the range of -1 and 1. Again, for the upper body and inter-leg desired angles output torque was split evenly across both hip joints, whilst the remaining desired angles (stance knee, swing knee, stance ankle and swing ankle) only required the torque to be applied to a single joint to adjust the angle. Seen in Figure 4.4, my re-implementation was also configured with identical body parameters to the original work. The walkers consisted of 9 sections - two hips, two thighs, two calves, two feet and an upper body. These were linked together with 8 joints, of which, the hip, knee and ankle joint pairs were actuated. I used an identical genetic algorithm to evolve network weights. Solomon *et al* modelled their genotype as the network weights across every neural network controller, initialised between 0.05 and -0.05. Every generation, the genetic algorithm simulated the population and sorted it by fitness, replacing the population (save the top performing elite individual in 2D simulations) with offspring produced through either mutation from the top 20%, random crossover from the top 20% or both. The mutation rates for each neural network controller were also evolved over the course of the simulation as an additional part of the genotype. The mutation rates were initially sampled from a normal distribution with a random mean between 0.05 and -0.05. Mutations were applied as the mutation rate multiplied by another random normally distributed value; the result of which was then added to the genotype. The control scheme I used matched the simplest style from the original, the Local Proportion, featuring only angle and derivative inputs relative to the desired angle for each neural network controller. The simplest control layout was chosen, only connecting the specific inputs for each desired angle controller, in order to simplify the task and reduce computation. In addition, a simpler task would be easier to scale up to a more complex one later on. When simulating in full 3D I added hip adduction and passive ankle joints, and an additional neural network controller with inputs for the new hip spread desired angle (torque was applied evenly across both lateral hip joints to adjust this angle), as in the original work.

The following are re-implementation changes. For reduced computation, walkers were given a lifetime limit of 120 seconds. This was to allow for many runs to be executed in a shorter timespan. Whilst this tradeoff might lose some of the advanced behaviour in later generations, I feel the diversity of genotype provided from more runs will be more useful. Fitness was also changed slightly, now modelled as distance travelled forwards in X divided by walker leg-span. This was chosen to give a more aesthetically pleasing representation of the length of a walk in steps, even though it

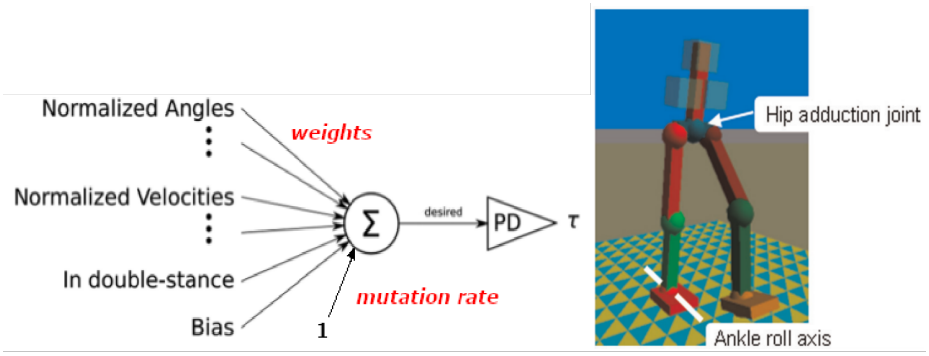

Figure 4.1: **The network topology and biped designed by Solomon *et al.*** (left) Their networks featured a simple summation of inputs, multiplied by their respective evolved weights, alongside an evolved mutation rate, to produce a torque value for each desired angle. (right) The model produced by Solomon *et al* can be seen with labelled lateral joints walking on rough terrain [77]. It also featured six sagittal joints at the hips, knees and ankles.

required a linear transformation to compare with the original fitness values. Finally, the masses on the thigh and shank segments were attached as separate blocks in order to be moved around later in an extension of the task; due to their positioning and additional scaling parameters, they are functionally identical to inbuilt masses in ODE.

4.2.3 Control Enhancements

The following are experimental changes. In order to skip the initial 2D bootstrapping used by Solomon *et al* and evolve the system in a 3D physics engine from scratch, several control-based enhancements were utilised. The principle enhancement was removing the control cost penalty after a given amount of time. As in the original system, walkers were given a lifetime torque allowance of 3000 units across all their joints to encourage efficient gaits. If this was exceeded at any point, the walker immediately failed. I stopped enforcing the limit as a failure condition after 2.2 seconds. This was chosen to allow the walkers to move more freely at the cost of lower efficiency, with the hope of producing increasingly complex, high-fitness behaviours. Furthermore, the 2.2 second delay was chosen in order to allow walker agents to transition efficiently into a walking cycle first from a standing position, before freeing up motion, as this initial phase was more complex. The value of 2.2 was chosen as half of an average lifetime length value of 4.4, after running several test walkers without the enhancement. An alternative method could have been to remove the limit across the entire walker lifetime, but this would have potentially disrupted the initial phase

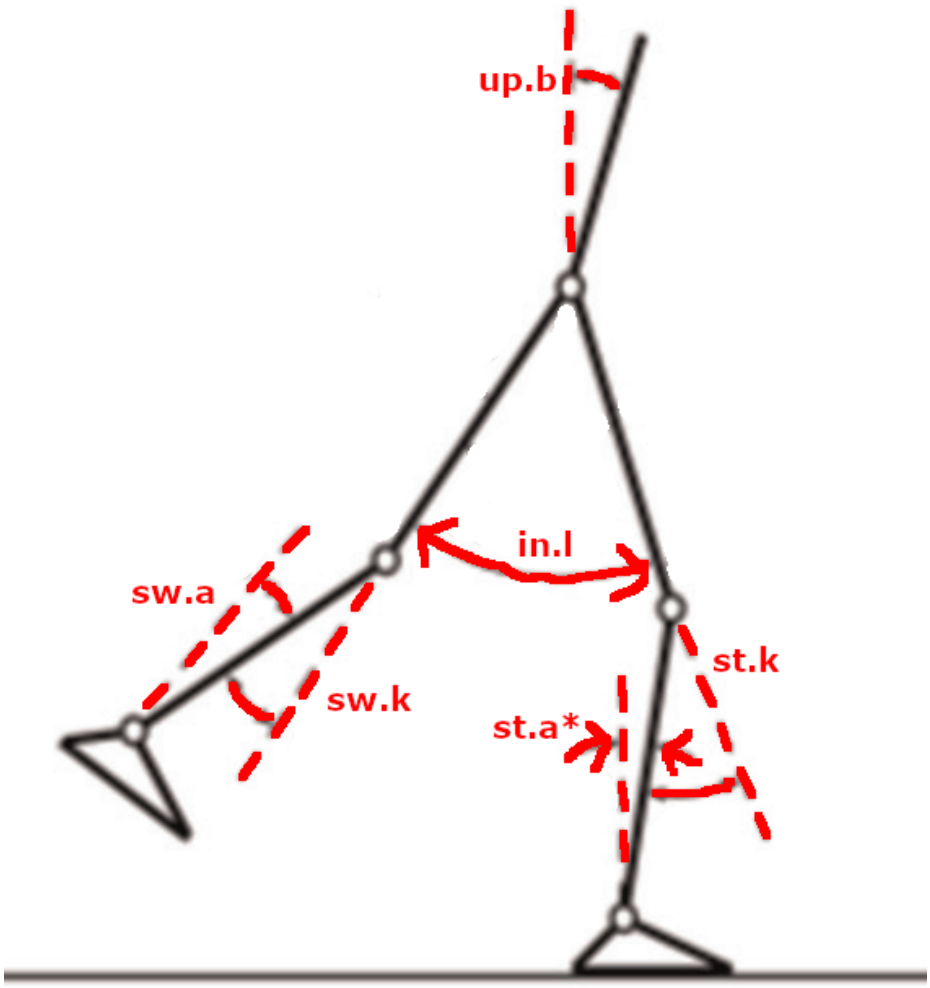

Figure 4.2: **Desired angles in 2D.** Solomon *et al* chose six desired angles for their 2D model. By applying torques to the relevant joints, they adjusted these angles to produce motion. Seen in red, the 2D angles chosen were the upper body, inter-leg, stance knee, stance ankle, swing knee and swing ankle angles. Note that the stance ankle angle was inverted. In 3D, an additional lateral inter-leg desired angle was added.

mentioned above. Another alternative could have been to find a more precise measure of when the transitional phase ended, but this would have required a thorough analysis of joint forces over time, and with the high number of time-steps, the high difficulty of the task, and high variation in agent behaviour, this would have been very computationally intensive.

The next enhancement involved a flight phase. In walking behaviours, a flight phase is defined as no part of the agent being in contact with the ground, hence flight. In the original work a flight phase was not allowed, to encourage walkers not to hop or jump forwards. I allowed a flight phase throughout walker lifetime instead. This was chosen again in order to free up more motion for potentially more complex gaits. An alternative could have been to only allow the flight phase after the initial phase, similarly to the control cost, but I felt that leaving the ground immediately could allow the walkers to fall down into a position to start cyclic motion with greater ease.

In addition, another aspect of the original system was the final section of the Linear Reactive controller neural networks, the PD Control. In the original work, these took in a desired output value produced from summing the inputs multiplied by the weights, and applied error correction proportional to the previous error and previous error derivative to produce a final torque value, which was then applied to the joints that would change the desired angle designated to the controller. Unlike the original, I chose to remove PD controls from the networks. I used ODE's inbuilt error correction, setting the desired output values as the velocity parameters for the corresponding hinge joints, instead of applying the torques directly. I chose to do this as it simplified the computational process by allowing more to be left up to the physics engine instead of introducing more parameters to the system. A potential alternative would have been to set the torques manually based around the desired output values without a PD control, but ODE can become unstable easily when too much force is applied to a body, which meant such a method would require extreme precision, and likely further complications.

Finally, the original system retains the best agent for elitism in 2D simulations. This allows selection to always move forward and evolutionary progress to not be lost to mutation or crossover. I changed this, introducing the usage of the top 15% being retained as elites, in both 2D and 3D. I chose this to allow for easier travel through the fitness landscape by maintaining a larger portion of successful agents to mutate from

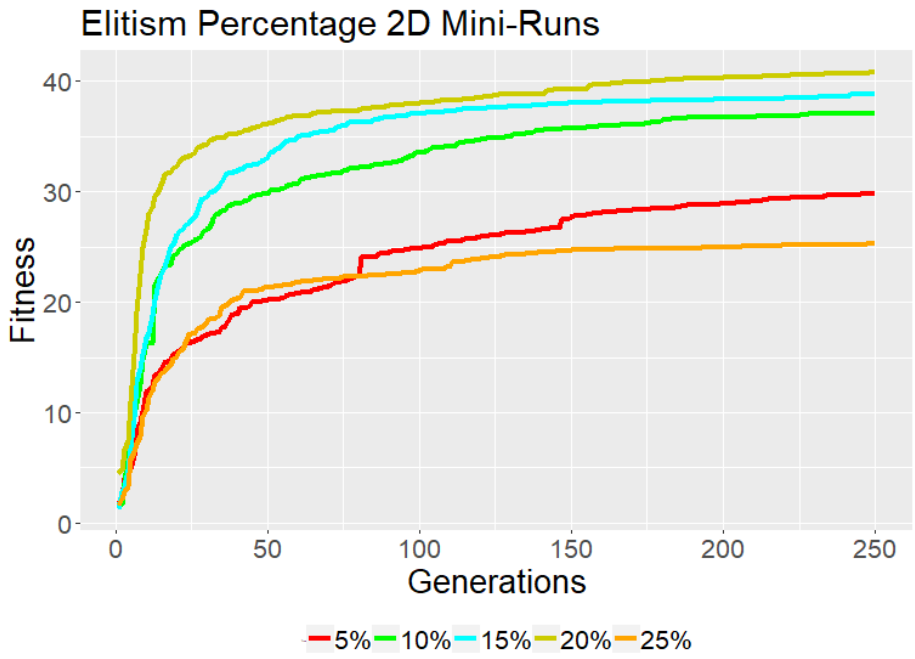

Figure 4.3: **Testing different elitism percentages.** Five percentage levels of elitism are tested in smaller mini-runs in 2D, in order to choose a final value for elitism in the enhanced system.

each generation. 15% was chosen after a series of mini-runs with a smaller population of 100 for only 250 generations (the standard in this work is 150 for 500 generations) were executed in 2D, with varying percentages from 5 to 25, shown in Figure 4.3. 5 to 25 was chosen as a range of five evenly spaced values in the upper 25% of the population. I feel having any more than 25% of a population labelled as elites and replacing less than 75% of the population slows down evolution too much, so this was chosen as the upper limit. Figure 4.3 shows the fitness curves from these mini-runs, with 10, 15, and 20 percent being distinctly separate from 5 and 25 percent, showing a peak in maximum fitness. As such, I chose 15% as the middle value of this upper class. My winner was also not selected from the elites of several final generations; instead the fittest of the final generation was chosen, as stochastic rough terrain was not tested.

4.2.4 Turning Behaviour

4.2.4.1 Turning Task

After forward walking, in order to achieve free locomotion in 3D, the ability to turn is required to move in lateral directions. This is therefore the most logical next step

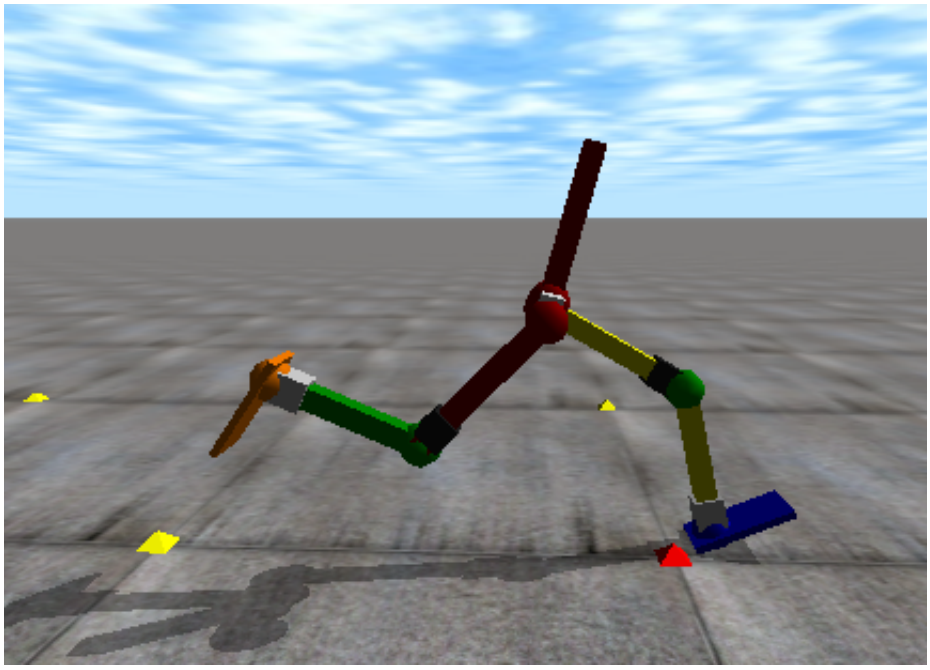

Figure 4.4: **My enhanced walker in ODE.** Highlighting the current stance leg in yellow during simulation, my re-implementation mostly matches the design of the original, with identical joints and morphological parameters. The differences to the original involved a reduction of control cost in the fitness function, increased elitism, a flight phase and removal of PD Controls. The remaining minor re-implementation changes are documented above. The blocks on the legs represent the locations of attached masses mentioned at the start of Section 2. It steps forward on flat terrain.

for behaviour after forward walking, and a good benchmark for my improved system. Using the turning work of Reil *et al* as a benchmark, I also investigated turning behaviours in the 3D walker agents produced by the improved system [64]. As in their work, these turning behaviours were defined as a laterally displaced goal point that the walker must reach from its starting point to achieve high fitness. Similarly to the standard 3D walkers, the turning walkers are equipped with hip adductor joints, passive lateral ankle roll joints, and feature a new neural network controller for an inter-hip desired angle. Whilst Reil *et al* used lateral sensors and a signal source to produce turning behaviour in their walkers, I instead rely solely on fitness function changes. Previously basing fitness on distance travelled forwards, I now base it around distance from a laterally positioned goal point, to encourage turning behaviour. Defined in Equations 4.1, 4.2 and 4.3, fitness is redefined as a maximum distance value minus the current distance of the agent to the goal point. In Equation 4.1, modelling xg and yg as the goal point coordinates, the maximum value is defined as twice the initial distance from the walker to the goal; this was chosen in order to maintain a positive value in most circumstances. Equation 4.2 models the standard pythagorean calculation of distance d, modelling xc and yc as current walker position. In addition, walkers received a bonus of 50 to fitness once they are within 2 simulation units of the goal point. This is demonstrated in Equation 4.3, modelling f(a) as the fitness of walker agent a. This was chosen in order to provide a vast difference in fitness between agents that had gotten close to the point and those that had not, encouraging turning further. High-fitness agents typically scored around 20 without the bonus. I chose to use this fitness function methodology as it provides a simple task for the walker to evolve without complexifying the neural network controller. An alternative could have been lateral sensors and a signal source, as in the work of Reil *et al*, but this would complexify the neural network controller with an additional input. Another alternative could have been to modify the environment to guide the walker into turning behaviour passively, but this would require adjustments to the walker morphology, not currently designed with passive dynamics in mind.

$$M = 2\sqrt{x_g^2 + y_g^2} \tag{4.1}$$

$$d = \sqrt{((x_g - x_c)^2 + (y_g - y_c)^2)} \tag{4.2}$$

$$f(a) = \begin{cases} M - d + 50, & d < 2 \\ M - d, & Otherwise \end{cases} \tag{4.3}$$

4.2.4.2 Parameter Search

In the CPG based evolved walker of Reil *et al*, the points [10,3] and [10,-3] are chosen as goal points for the signal sources [64]. No reason is given for this decision, but as the choice of goal point requires differing turning arcs, here I examine several goal point pairs in order to choose an ideal pair for extended testing. I examine 25 y-positive/y-negative pair permutations, using the combination of x-coordinates [6,8,10,12,14], and y-coordinates [1,2,3,4,5] in both positive and negative. I run each of these pairs 10 times to assess behavioural quality. Quality is assessed through two metrics: final distance to goal point and arc roughness. Final distance to goal point is defined as the distance between the goal point and the walker agent at moment of failure, either through a fall or from running out of time. This is calculated identically to Equation 4.2 in the previous section, modelling xc and yc as current position, and xg and yg as goal position. The other quality metric I use here I refer to as arc roughness. Seen in Figure 4.5, by taking the gradient between the walker's positions at each time-step and summing the differences between each gradient and the next, it is possible to achieve an estimate of the smoothness of the walker's turning arc. This is specified more precisely in Equation 4.4, modelling x and y as walker position, t as current time-step and T as total time-steps in a walker's lifetime. Walkers with a more varied or rough trajectory will have a greater local difference amongst arc gradients. However, this method relies on the assumption that a smooth arc is the ideal trajectory, something that may not be indicative of behaviour. Equally, final distance from point may also not indicate quality of gait. It is for this reason the two metrics are combined together. Each was chosen for simplicity due to the large number of permutations tested. A possible alternative to distance to the goal could be to take total distance travelled in the lateral dimension, but this would require additional normalisation to scale the values, and could give similar scores for vastly different trajectories. A possible alternative to the arc roughness measure would be to fit a polynomial spline function to approximate each turning arc and then measure smoothness via the polynomial function's derivatives instead. However, this would require a large deal of estimation as opposed to taking the values directly from the simulation.

$$\sum_{t=1}^{T-2} \frac{y_{t+2} - y_{t+1}}{x_{t+2} - x_{t+1}} - \frac{y_{t+1} - y_t}{x_{t+1} - x_t} \tag{4.4}$$

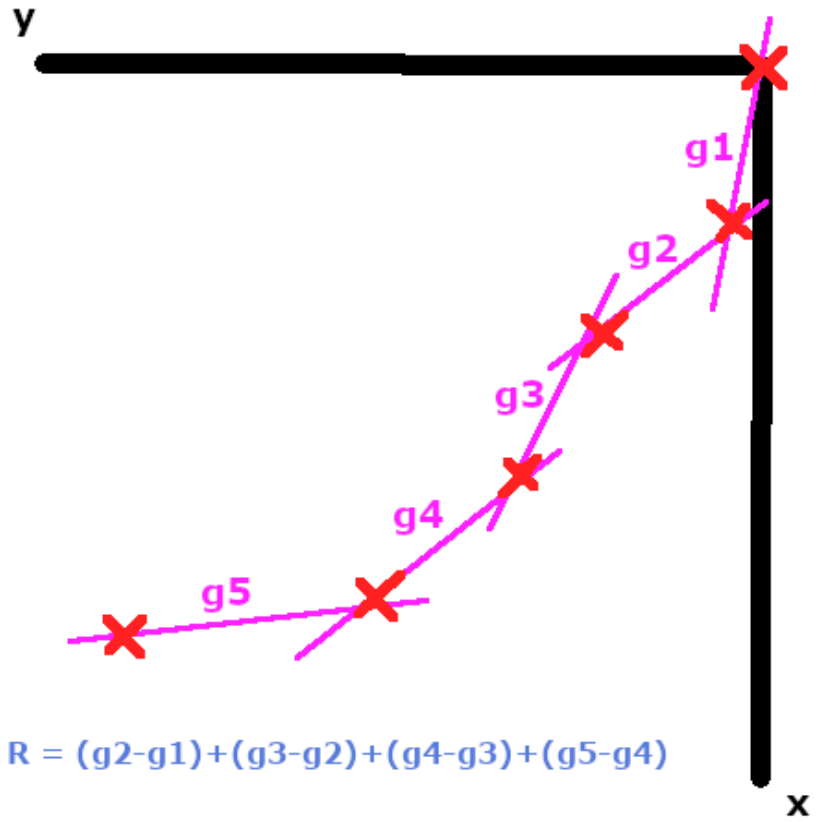

Figure 4.5: **An example of arc roughness calculation.** Viewed from above, with red Xs representing walker position per time-step, gradients g1 to g5 are calculated and a final roughness value R is produced for the walker trajectory by summing the local differences between them.

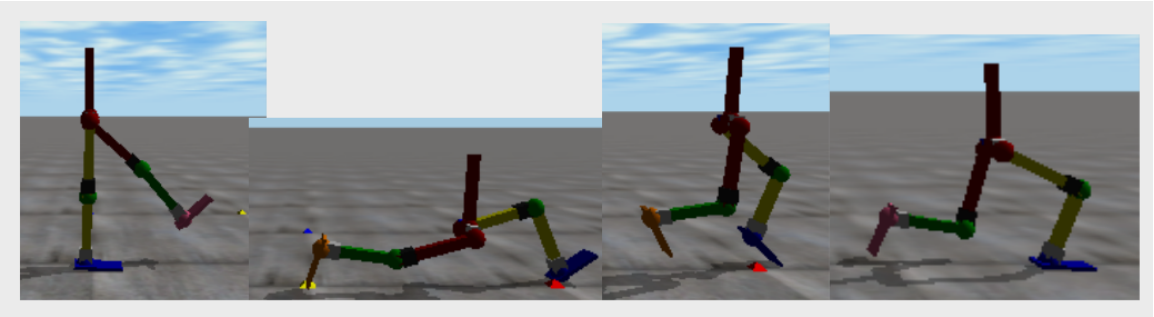

Figure 4.6: **An example of 2D constrained walker gait.** The 2D walker steps forwards, transferring weight from one foot to the other.

4.3 Results

4.3.1 Walking in 2D

After applying the control enhancements and elitism changes, runs were executed in both 2D constrained and 3D, to compare with the original work. Figure 4.8 shows the fitness curve for the 2D constrained runs in blue. Each run contained 150 walkers run for 500 generations, chosen as the largest number of generations for a large enough population that could be executed within a reasonable runtime. Twenty-five of these runs were executed for each average fitness curve. This value was chosen as slightly over 20, the generally accepted minimum sub-group sample size for quantitative research. The 2D fitness curve ascended rapidly in the first 50 generations, after which it slowed before reaching a final value of 30 steps. Figure 4.6 shows an example of a constrained 2D gait. Walkers were initialised with the left leg up to produce cyclic motion easier, as in the original work. The gait began in the standard position with a leg raised, before falling onto the leg, pushing up and falling again, stabilising into a walk cycle. The knees remained static throughout the gait. Compared to the original, the walkers were able to travel a similar distance.

4.3.2 Walking in 3D

Figure 4.8 also shows the fitness curves for the walkers in 3D, with lateral actuation enabled. Walkers were again initialised with the left leg up to produce cyclic motion easier. The 3D walkers only managed a fitness of around 8 by generation 500. This made for a stark 75% drop in average fitness compared to the 2D constrained walkers. The 3D walkers had a similar gait to the 2D walkers, stretching the stance knee to push forwards, but also made use of their passive ankle joints and actuated hip adduction joints to stabilise themselves laterally. Despite this, the 3D walkers quickly

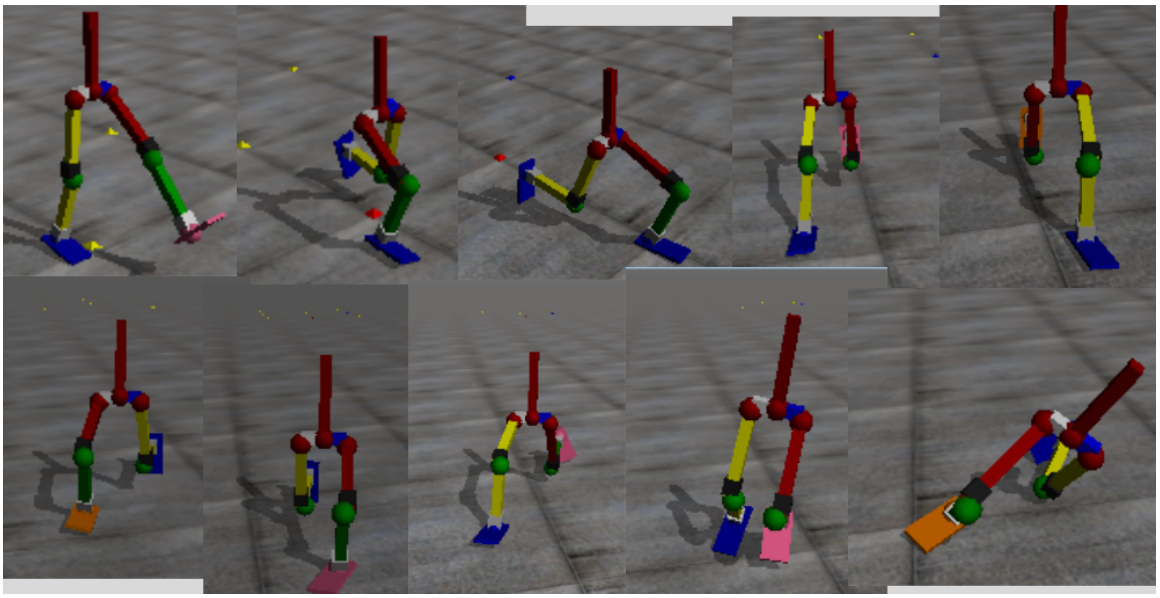

Figure 4.7: **An example of a 3D walker gait.** The walker can be seen stepping forwards multiple times before falling sideways.

succumbed to cumulative error in this plane and fell sideways after walking a short distance. Compared to the original work, the 3D walkers also did not travel as far and could not walk on rough terrain. However, they still travelled around 8 steps; this demonstrates the stability and success of their gaits. They also did not require prior processing in a 2D engine.

4.3.3 Turning

The results of the turning parameter search can be seen in Figure 4.9. Each goal point permutation pair was tested with 10 runs in both the positive and negative direction. The average final distance to goal and arc roughness of each pair was then calculated and normalised between 0 and 1 to match in scale. The perturbations with higher x coordinates are shown to have a higher average distance to goal upon failure. This can be assumed to be because they have further to walk forward as well as turn, reducing the likelihood of success. The arc roughness, on the other hand, does not follow any such trends. Finally, compared to the final distance values, most of the arc roughness values are low, suggesting a smooth turning arc is easy to produce with this task. This makes sense, as only one turn is required. After consulting the combined metrics, I made the decision to investigate the [8,4] [8,-4] goal point perturbation further (y4x08), as it was the widest turn with the longest wavelength to have both

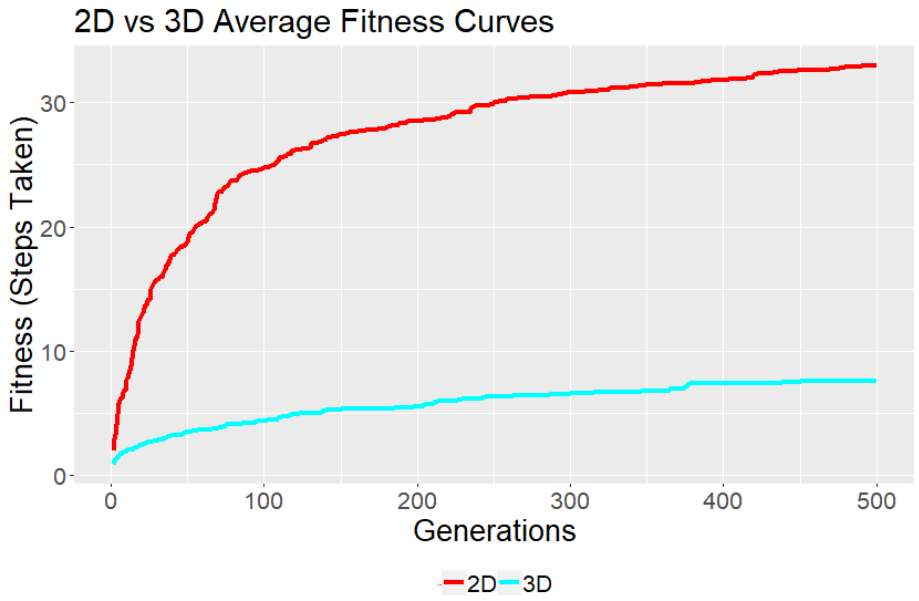

Figure 4.8: **The 2D constrained and 3D fitness curves.** The fitness curve of the 2D walker runs and the fitness curve of the 3D walker runs in steps taken. Twenty-five runs were executed for each average fitness curve. The 3D walkers perform considerably worse than the 2D.

metrics at a similarly low level, below 0.125.

Figure 4.10 (right) shows the paths of the two fittest agents in the population for the fittest (out of sets of 25) left and right final permutation turning runs, compared with those of Reil *et al* (left). As per the previous parameter search, these runs featured goal points of [8,4] and [8,-4]. The left and right runs featured different single turn trajectories. The left turn smoothly turns toward its goal over time, whilst the right turn initially has a more straight path, turning later into its lifespan. The right turning run did not reach its point, although it did appear to begin a turning motion before falling, and still successfully turns to [8,-2]. I suspect this may be due to the nature of the fitness function; by applying a large bonus when walkers reach a distance of 2 from the point, walkers are less incentivised to travel any further towards the point once they reach this boundary. A scaling value would have been a better decision. In addition, the walkers are initialised with the left leg up to produce cyclic motion easier. This could have made it harder for the walker to turn right from its initial position. The trajectories produced are also similar to those produced by Reil *et al* seen on the left side of the figure, with mine being slightly smoother. Figure 4.11 shows the gait for the left turning run. It firstly walked forwards with

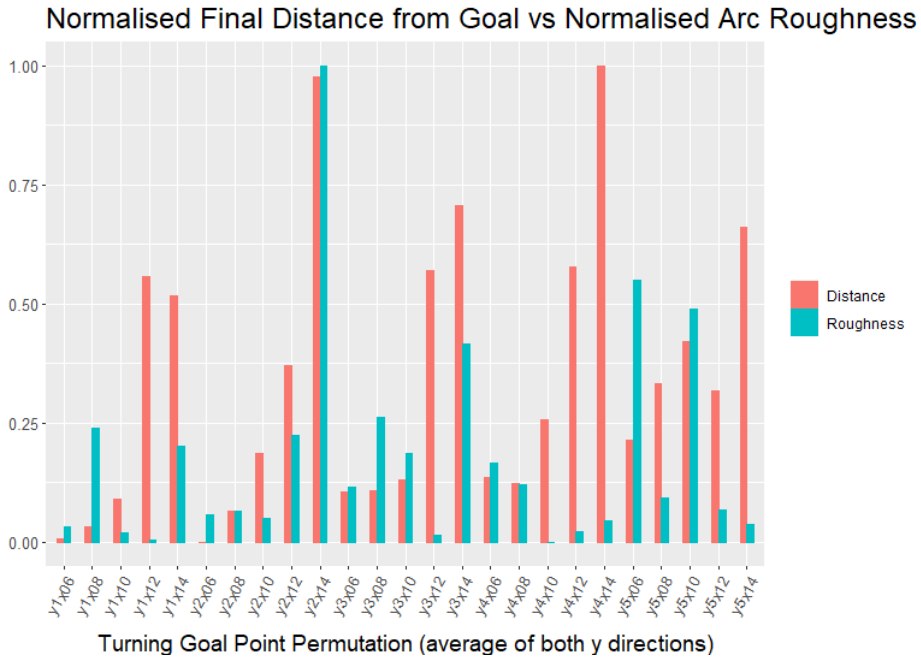

Figure 4.9: **Turning perturbations tested for each quality metric.** The normalised final distance and normalised arc roughness for each perturbation pair, averaged across the positive and negative y directions. A permutation label of y1x06 represents the average of goal point pair [6,1] and [6,-1]. This rule then applies for the other permutations.

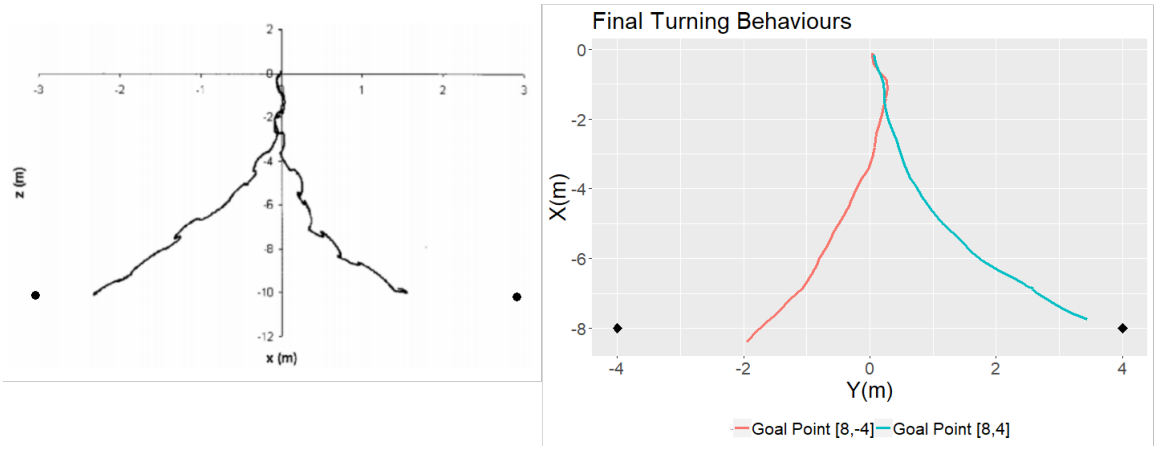

Figure 4.10: **The left/right turning paths from the work of Reil *et al* and this work.** (left) The left/right paths followed by the lateral sensing walkers from their work, looking to reach points (10,3) and (10,-3), highlighted as black circles. (right) The paths followed by the fittest left and right turning walkers I produce here from the two final sets of 25 runs, from the origin to their goal points at (8,4) and (8,-4), highlighted as black diamonds. They modelled the forward direction as Z and horizontal as X, whereas I modelled them as X and Y; there is no functional difference between the two styles.

a similar gait to previous runs, then utilised its hip adduction to start adjusting its heading toward the goal point. Figure 4.12 shows the gait for the right turning run. It followed a similar transition, but fell forwards before turning enough to reach its goal point.

4.4 Contributions

The contributions made by this work are as follows:

- By introducing control enhancements to the Solomon *et al* Linear Reactive system (removal of torque limit after a given time, enabling a flight phase, replacement of PD controls with ODE methods), alongside an additional 15% elitism, I demonstrated they enabled the evolution of fully 3D walking behaviours within the system, without any prior 2D bootstrapping [77]. The enhanced system they produce is simple enough through the Linear Reactive controller network and the lack of bootstrapping required that it can serve as a basis for future bipedal works. This matches my Chapter aim to remove bootstrapping and my general aim to increase walker fitness.

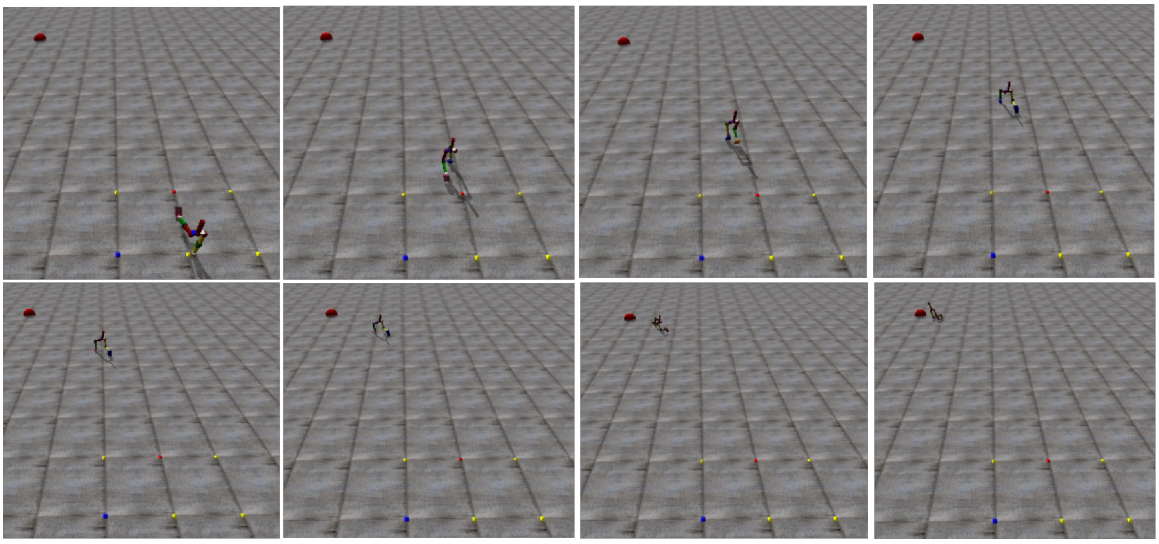

Figure 4.11: **The best left turning gait.** The most successful left turning gait from the final set of 25 runs. It steps forward, adjusting its heading towards its goal point as it moves, eventually reaching it.

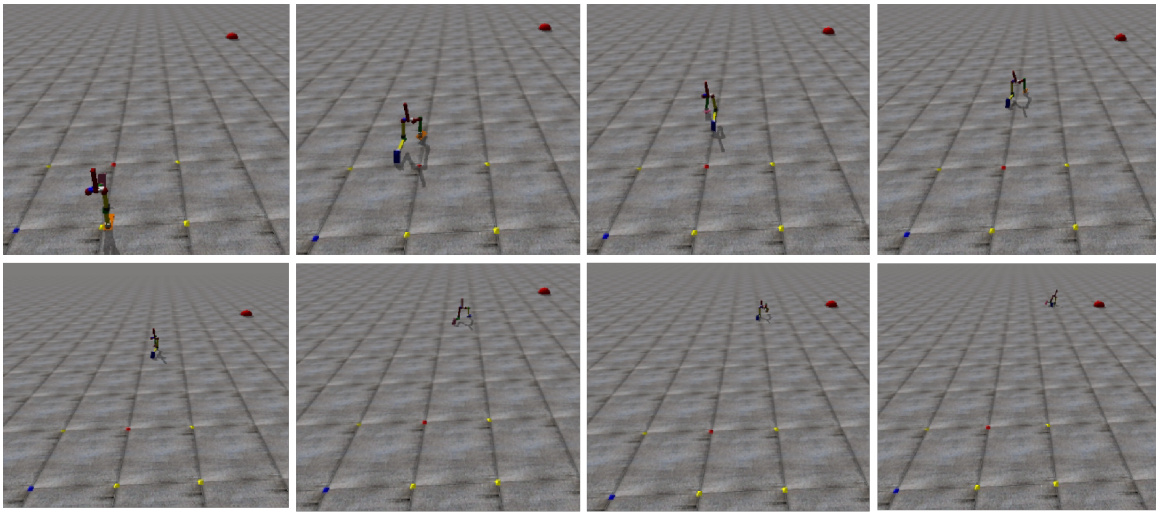

Figure 4.12: **The best right turning gait.** The most successful right turning gait from the final set of 25 runs. It also steps forward whilst adjusting its heading towards the goal point, but falls having not turned enough.

- I demonstrate the control enhancements enable the production of successful life-like 2D constrained gaits in the enhanced system, taking as many as 30 steps and matching the original system in fitness.
- I demonstrate the control enhancements enable the production of 3D gaits capable of as many as 8 steps in 3D in the enhanced system, although these were not as successful as the original system.
- I demonstrate the control enhancements enable the production of single left-right turning behaviour in the enhanced system by applying a new distance-based fitness function. This behaviour was comparable to that seen in Reil *et al* [64]. This matches my Chapter aim and general aim to produce complex behaviour.

4.5 Discussion

A possible implication of the results here is the role of fitness functions in enhancing evolutionary systems. The control enhancements feature an adjustment to a fail condition of the walker task, whilst the single turning experiments produce a new fitness function based around distance to a target point. Both these changes had a notable effect on the walker system, removing the need for prior bootstrapping and facilitating the evolution of single-turning behaviour respectively. This can be interpreted as evidence that fitness function enhancements are the most important aspect to consider when enhancing an evolutionary system. If this were the case, it could help to focus the philosophy of further evolutionary works towards the evolution of complex behaviour. It could also suggest alternate investigations for previous experiments that had not utilised this philosophy.

Another implication of the results here is the success of the "Novelty over Efficiency" philosophy described previously. This novelty can be useful for the bipedal walking task, particularly as balance is a constant requirement, and an agent capable of several novel recovery methods would exhibit increased robustness. By reducing the penalties for efficient motion in the system and allowing for a greater number of elite agents in the population, the walkers were free to move with more force and in previously unseen ways. This can be interpreted as evidence for the application of this philosophy to other works. This could lead to the exhibition of complex behaviour in other evolutionary works, exhibiting a more diverse population with a larger variety

of successful agents, which could in turn have a higher chance of producing complexity, especially in 3D balancing tasks. Again, this could also provide an alternate line of investigations for existing works.

A final implication of the results here is the potential of incremental evolution on the system. In the original work of Solomon *et al*, the simple neural network controllers are bootstrapped through several configurations to produce the final 3D walkers. This technique is often seen in evolutionary systems and referred to as incremental evolution, in which a larger fitness task is broken down into sub-tasks, with the final population in one sub-task used as the initial population for the next. One of the perks of my enhanced system is its simplicity of neural network controller and lack of prior bootstrapping. This simplicity could make it the ideal candidate for further incremental experiments. This could lead to more complex behaviour, such as the turning behaviour being bootstrapped into a behaviour involving following a moving point or avoiding an obstacle via a second sub-task. This could also lead to the introduction of passive-dynamics into the system as a sub-task, such as the gravity-powered walker morphology seen in McGeer [51]. This would lead to complexity through morphological replacement of computation, freeing up neural network controller-space for other aspects of behaviour.

4.5.1 Acknowledgements

My code is based on the work of Solomon *et al*. It is available via GitHub at *https://github.com/benjack795/solomon-reimp* for exact specification and future reproductions.

Chapter 5

Neuroevolution of Humanoids that Walk Further and Faster with Robust Gaits

5.1 Chapter Aims

- After enhancing a leading bipedal system with a notably simple neural network controller scheme, in this Chapter I aim to enhance another leading bipedal system from Salimans *et al* with fitness function modifications to further test the robustness of my methods in a context comparable to the reinforcement learning seen from Heess *et al* [68] [35]. This is in line with my aim to increase walker fitness.
- I also aim to produce robust gaits more life-like than the "shuffling" behaviour exhibited by the majority of the gaits produced by the Salimans *et al* system. This is in line with my aim to produce complex behaviour.

5.2 Methodology

5.2.1 Previous System

The enhancements featured here are applied to the Salimans *et al* evolution strategy system, via code produced by Such *et al* for their work [82], which was already capable of producing gaits that move 3D Humanoid-v1 agents quickly and efficiently enough to pass the humanoid walking task. The full code is referenced at the end of this Chapter. The humanoid walker's goal is to travel (in any direction) as fast and efficiently as possible, failing when its torso falls below (or above) a certain height. Being exact, fitness is defined as the sum (over time) of four rewards/penalties that are

computed at each timestep: a reward for linear velocity, a control cost based on energy expended, a cost based on how hard the humanoid impacts the ground, and a reward for standing. I ran each permutation of enhancements (and the default base-case) 20 times, each for 600 iterations (generations). I chose to use the Salimans *et al* evolution strategy version due to its superior results, when compared to the Genetic Algorithm version produced by Such *et al*. The architecture maps 376 inputs (humanoid state variables: position, rotations, velocities, forces and inertia values) to 17 joint torques - approximately 30k parameters are evolved in between as the genotype (network weights and biases). They used a natural evolution strategy, modelling population as sets of network parameters produced with hyper-parameters. Each generation, the strategy evaluated a number of random perturbations of the parameters produced by the hyper-parameters. The results were then combined, the gradient estimated and the hyper-parameters updated for the next generation. The physical body of the model can be seen in Figure 5.1. Each set of 20 runs took around three days to evolve using 40 CPU cores. Each episode (walker fitness evaluation) was limited to a maximum of 1000 time-steps for reduced computation.

5.2.2 Fitness Function Enhancements

In the Salimans *et al* system, they define fitness as the cumulative sum of four terms that are computed at each time-step. The fitness function produced by Salimans *et al* is modelled in Equation 5.1 as fitness F, with T being the total time-steps across a walker's lifetime. The first of the terms in the function is a reward for linear velocity at time-step t, modelled as v. This term is calculated as the distance travelled forwards by the center of mass in the x dimension since the previous time-step. This was chosen as a positive term to give the walker a higher fitness from travelling further forwards, encouraging locomotion. The second term, modelled as c, is a control cost penalty per time-step. This is calculated as the sum of the squared torque values applied to each actuator. This was chosen as a negative term to penalise the walker for using too much force to control itself, enforcing efficiency. The third term, modelled as i. represents a penalty for impact cost. This is calculated as the sum of the squared external forces on each body. This was chosen as a negative term to penalise the walker for smashing into the ground too hard, in order to prevent jumping behaviours and focus on walking behaviours. The final parameter in the function, modelled as a, is a bonus for remaining alive. This is a static value applied to the rest of the function scaled to the other values and was chosen as a positive term

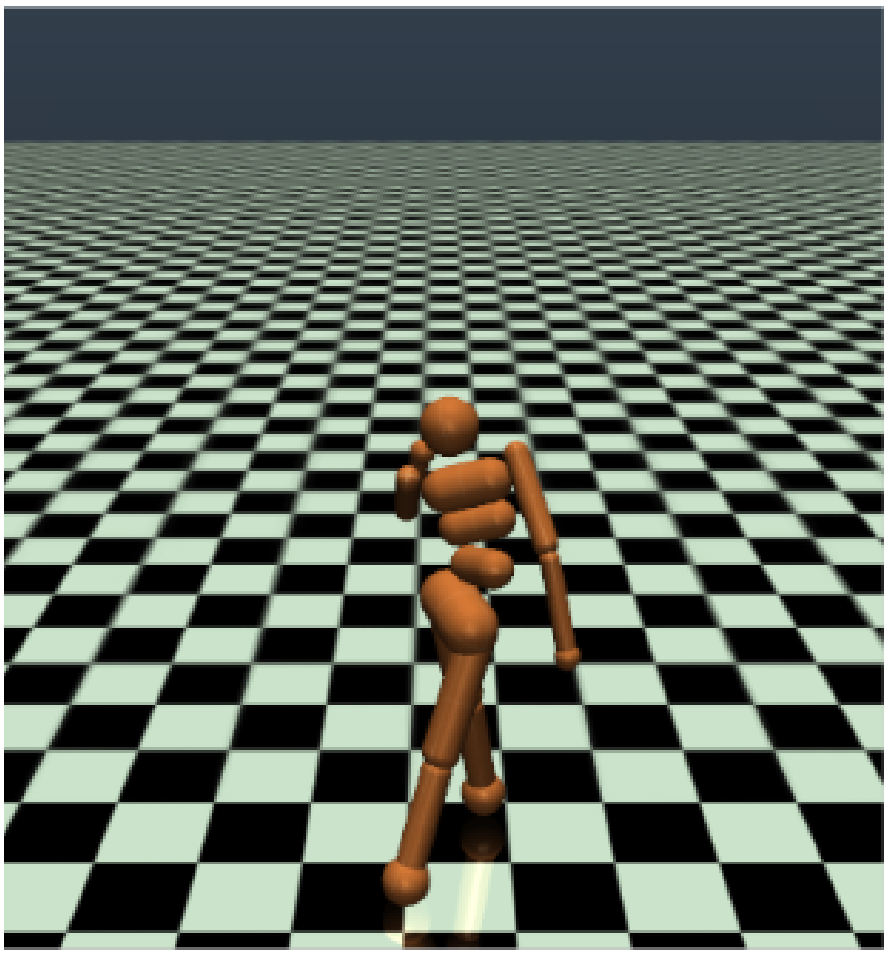

Figure 5.1: **The Humanoid-v1 walker used in this and previous Mujoco-based work.** The humanoid in the humanoid-v1 environment must move forwards without falling.

in order to make sure the walker is always receiving a positive value for continuing despite the other penalties.

$$F = \sum_{t=1}^{T} v_t - c_t - i_t + a \tag{5.1}$$

Even in the default case, without any enhancement, this led to the evolution of fast, efficient walking in line with previous reinforcement learning results. However, a considerable number of evolved gaits involved a non-life-like shuffling motion. These walkers slid along the floor with small movements of their feet.

5.2.2.1 Control Cost Enhancement

The first enhancement I employed involved reducing the control cost within the fitness function. In the previous Chapter, the Solomon *et al* system employed a torque limit across walker lifetime to encourage efficient walking. In order to re-implement their system without prior bootstrapping, I removed this limit after a hand-picked time period to allow evolution to be less constrained. As such, I employ a similar method here, sacrificing efficiency for novelty by reducing the value of the control cost penalty in the fitness function. To encourage gaits that use more motion here in novel ways, I applied a scalar multiplier to the control cost term, allowing for new behaviours at the cost of generating less efficient walkers (when the multiplier is below 1). This multiplier could be applied throughout each simulation episode or from a set timestep during each simulation episode. Delaying was be beneficial as the gaits evolved typically had a "catch" phase in which walkers aligned themselves from the starting position into cyclic motions. As this catch phase required more carefully considered motion, a reduced control cost here could produce larger motions that would disrupt the balance of the transition to a cyclic gait. The aim of this first enhanced set of runs was to produce novel gaits with longer walks by reducing the control cost in the fitness function. Parameter rationale for this enhancement can be found in the results section.

5.2.2.2 Balance Enhancement

The second enhancement I employed was a fail condition involving the balance of the walker. In the original system, balance is described as the walker's torso's vertical (z) component being outside the range of 1-2 simulation units. To improve walkers' fitness I introduced an additional constraint for the x and y-dimensions, terminating walkers with less upright postures. If the torso's center of mass moved outside a circle

centered at the midpoint between the walker's two feet (each projected down to the ground plane) then it was considered a failure. The circle's radius is set as a multiple of the current distance between the midpoint and (either) foot. The radius-multiplier can be adjusted to give a more or less tight constraint. The aim of this set of runs was to produce novel gaits with longer walks through stricter balance enforcement. Parameter rationale for this enhancement can also be found in the results section.

5.2.2.3 Combined Enhancement

The two enhancements were also combined, using the most successful parameters (control-cost multiplier and its delay, and balance-circle radius-multiplier) for each. The aim of this was to test if such a combination would achieve superior results.

5.2.3 Rationale

Fitness function enhancements and failure conditions are investigated here as a key methodology for improving behaviour. Fitness function enhancements were chosen as they can be seen as the most direct way to influence behaviour in an evolutionary system. This was seen from the previous Chapter, where a fitness function was changed to produce new turning behaviour with the same agents. This can be seen from Azarbadegan *et al*, who added a damage metric to the fitness function of their work to produce bipedal agents in Sims' Virtual Creatures system [2]. Failure conditions are also investigated here for this reason. More specifically, the control cost enhancement was chosen after the results seen from the previous control cost modifications were shown to improve system behaviour enough to not require prior bootstrapping, thus achieving similar results by sacrificing efficiency for novelty. The balance enhancement was chosen as balance is arguably the most important part of any bipedal task, with potential fall failure at all times. In addition, with a fail condition already on the Z component of the center of mass, it was a simple and intuitive step to penalise this further. Alternatively, novelty search could have been utilised here in some form alongside fitness to encourage novelty [43]. However, this would have required a much more extensive parameter search to find the exact weighting between the two metrics to produce ideal behaviour. As another possible alternative, an additional sensor setup could have been introduced to track signals to stay balanced, similarly to the turning work of Reil *et al* [64]. However, due to the fully connected nature of the network, adding a sensor input would introduce a large number of additional parameters to the system, complexifying it.

This Chapter uses the Salimans *et al* evolution strategy system based in the Mujoco engine, despite the previous Chapter using an enhanced version of the Solomon *et al* system, based in the ODE engine. This could seem like a disconnect between works; however there was rationale behind this decision. Firstly, testing my enhancements across multiple types of bipedal system establishes their robustness. The Salimans *et al* system has a completely different neural network controller archetype, utilising a single deep network with thousands of parameters, instead of the set of simple perceptron networks seen in the Solomon *et al* system. In addition, the evolutionary process used is an evolutionary strategy, instead of the genetic algorithm used in the previous system. The system in this Chapter models population as a distribution of parameters as opposed to a literal population, and makes gradient updates every iteration instead of traditional selection. These differences highlight the rationale to test existing methodologies on a new system for robustness. Secondly, unlike the previous system, this system uses an environment and a task that has already been completed by existing machine learning methods. This provides an exact benchmark to test my results against, achieving my other aim - highlighting the benefits of evolutionary processes against that of reinforcement learning. Finally, the Salimans *et al* system can help me test how well my methods produce novel behaviours, in accordance with the novelty over efficiency rationale I discussed in the previous Chapter, and my general aim of producing complex behaviour in bipeds. One of the weaknesses in several behaviours produced by the default version of the Salimans *et al* system is the lack of traditional bipedal gaits. Walkers would shuffle along the floor like a wool carpet. This represents an opportunity to produce a more life-like gait in comparison to the default behaviours. An alternative to this would have been to test the support polygon follow-up methodology on the existing Solomon *et al* system, but this would be less relevant and provide none of the advantages listed above.

5.2.4 Robustness to Action Noise

The MuJoCo 3D Humanoid-v1 environment contains a parameter for the standard deviation (ac-noise-std) of Gaussian noise to be added to the actions taken by walkers. To test the robustness of the evolved gaits, I evaluated evolved walkers with noise levels (ac-noise-std) from 0 (no noise) to 1, to observe the degradation of each metric (average speed, distance travelled and episode length, or amount of timesteps before a walker fails) until the walkers no longer achieved (lengths long enough to be typical of) stable gaits. The aim of this was to test whether or not the combined enhancement

would result in more robust gaits, i.e. gaits with higher values in these metrics at higher levels of noise.

5.3 Results

5.3.1 Control Cost Enhancement and Parameter Search

I tested the enhancements to assess their affects on the speed, distance and duration of 3D Humanoid-v1 walks evolved using a replica of the Salimans *et al* evolution strategy system, produced by Such *et al* for their work [82]. All configurations discussed in this Chapter are executed in sets of twenty 600 iteration runs (the largest number of iterations that could be executed within a reasonable runtime). Twenty was chosen as the generally accepted minimum sub-group sample size for quantitative research. Figure 5.3 shows the results for the default (d, no enhancement) evolved behaviours and for walkers evolved with a control-cost multiplier of 0.25. In order to demonstrate notable results in this chapter, I utilise the Mann-Whitney U test for significance. This was chosen as the tests all featured the effects of a single change to the system on one continuous outcome value. As there were several tests performed in multiple groups, the Bonferroni correction was applied as a simple way to ensure there was little room for statistical error within the tests. When gaits are investigated, the darkness at the end of the gait figures occurs as the humanoids walk out of the range of the white floor texture. The floor texture and shadows have been altered from the original task environment for clarity.

The value of 0.25 was chosen for the control cost reduction multiplier value after a brief parameter search. Twenty 100 iteration mini-runs were tested at values 0.25, 0.5 and 0.75. This reduction was applied throughout the run here, to give an impression of which would be ideal to apply on a delay in the later experiments. These three test values were chosen as three evenly spaced increments at which to reduce the control cost. The control cost was not removed completely as my aim for this Chapter was to produce more complex life-like gaits, and too much motion could produce irregular ones. Figure 5.2 shows the average speed per run for each of the mini-runs for each of the parameter categories. Speed was chosen as the statistic to judge these mini-runs on as walkers with reduced motion that were still capable of higher speeds would be rarer and therefore more indicative of higher quality. The results show a fairly even speed split between the three parameters, ranging from 0.2 to 0.7 metres per second. p75 had the highest median and upper bounds, but not by very much.

As such, I decided to go with p25 as it would likely have the most impact of the three, being the largest reduction. To confirm this decision, I also directly compared the values between the groups. Using a Bonferroni correction for the two comparisons and reducing the significance margin down to 0.025, I observed that neither the p75 (Mann-Whitney U=191, n1=n2=20, p=0.40129, one-tailed, Bonferroni correction for 2 comparisons) or the p50 (Mann-Whitney U=178, n1=n2=20, p=0.27425, one-tailed, Bonferroni correction for 2 comparisons) were found to be higher than the p25 according to a statistical significance test.

I either applied the control cost multiplier throughout (xp25) or from timestep 150 (s150) or from timestep 500 (s500). As each of these were measured in three metrics (speed, distance travelled, and episode time) and compared against the default, I applied the Bonferroni correction for 9 comparisons, for a reduced significance margin of 0.0056. Timestep 150 was chosen as an estimate of the time at which the successful default gaits were reaching cyclic motion during preliminary runs. Timestep 500 was chosen as the halfway point for a full-length episode. Median speed (averaged over the time of each evaluation) in the s500 runs was more than twice that in the default runs, although not found to be statistically higher. Median distance travelled in the s500 runs was also more than twice that in the default runs, but not statistically higher. For episode time (amount of timesteps before a walker failed), all medians were the maximum value (1000, the walkers lasted till the end of the simulation) and I found no differences. Figure 5.6 includes high-fitness gaits produced by the default and s500 runs, without action noise. The default runs' gait (top-left) shows a shuffling behaviour based around the knee joints. The s500 runs' gait (top-right) also shows a shuffling gait using the knee joints, but, unlike the gaits produced by the default, the knees crossed over, putting one leg in front of the other. This improved gait may be due to reduced importance of keeping energy expenditure low (at least per timestep rather than per unit distance) once a walker had reached a cyclic motion.

5.3.2 Balance Enhancement

Figure 5.4 shows the results for the default (d) evolved behaviours and for walkers evolved with the balance enhancement, with balance-circle radius-multipliers 1.00 (r100), 0.75 (rp75), 0.5 (rp50) and 0.25 (rp25). These values were chosen as four evenly spaced percentiles of radius tightness for the balance constraint, scaling down from the maximum value and producing a smaller circle for the walkers centre of mass to remain within. Four permutations of 20 were chosen due to time and computing limitations

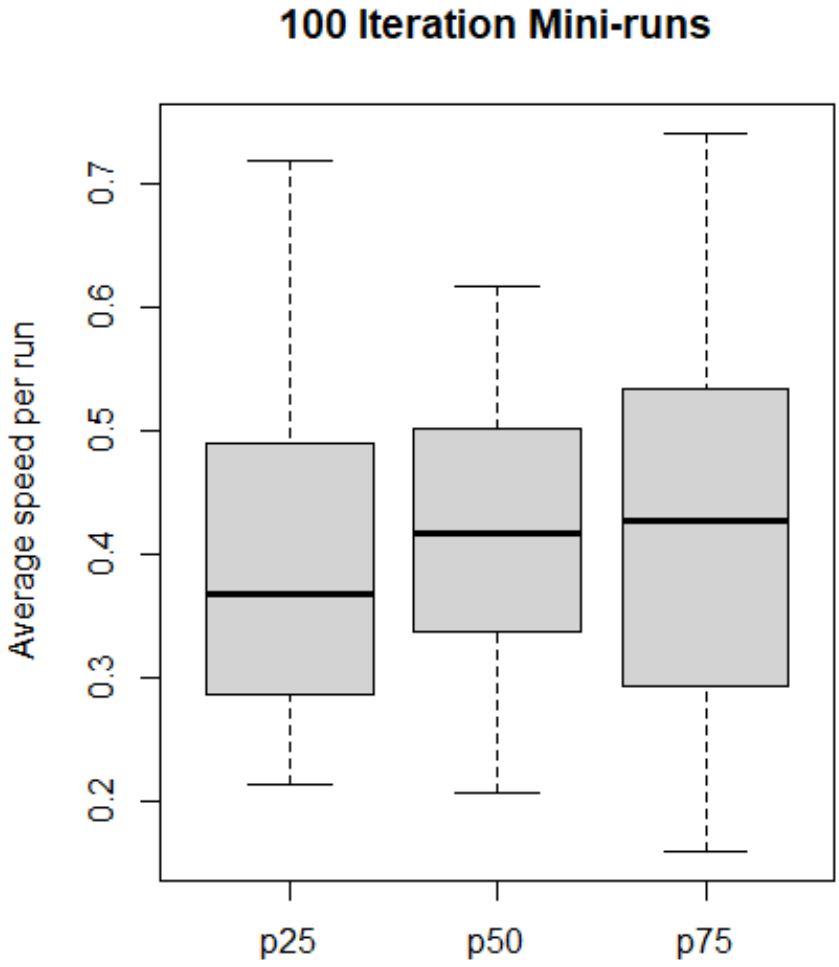

Figure 5.2: **Parameter Search Mini-Runs.** Average speed per run for the evolved behaviours for walkers evolved with the 0.25 control-cost multiplier throughout (p25), the 0.5 control-cost multiplier throughout (p50) and the 0.75 control-cost multiplier throughout (p75), for each set of twenty mini-runs of 100 iterations. Median speed and upper bounds were slightly higher in the p75 runs, but otherwise the results were similar, so the decision was made to use p25, after further checks revealed neither p50 or p75 were found to be faster than p25 according to a statistical significance test.

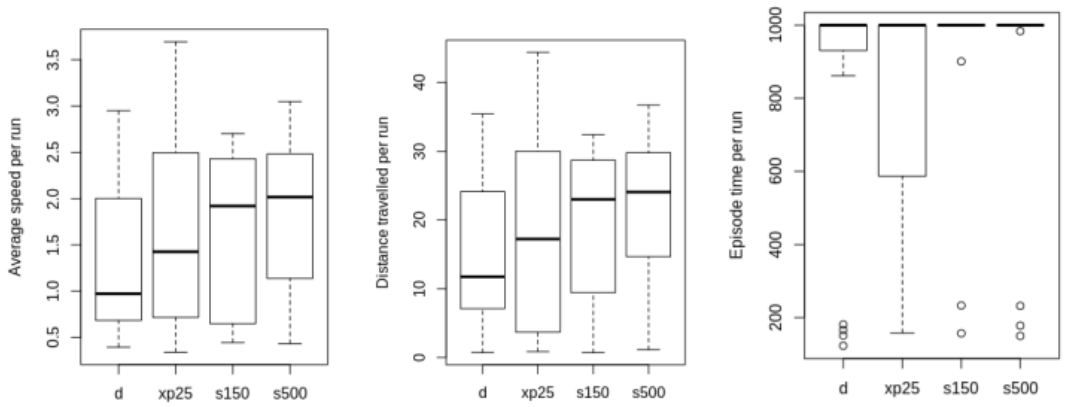

Figure 5.3: **Control Cost Results.** Results for the default (d) evolved behaviours and for walkers evolved with the 0.25 control-cost multiplier throughout (xp25), from 150 time-steps and from 500 time-steps (s150, s500): speed (left), distance (middle) and time (right) for each set of 20 runs. The s500 runs were found to have higher speed and distance travelled medians than the default. The s500 and s150 runs had the highest proportions of walkers reaching the maximum length of time allocated without failing, but neither were found to be greater than the default.

from the deep nature of the system and additional followup experiments. As each of these were measured in three metrics (speed, distance travelled, and episode time) and compared against the default, I applied the Bonferroni correction for 12 comparisons, for a reduced significance margin of 0.0042. Median speeds in the rp75 and rp50 runs were more than twice that in the default runs, with the former distributions found to be higher than the latter according to statistical significance tests (rp75 Mann-Whitney U=97, n1=n2=20, p=0.00264, one-tailed; rp50 Mann-Whitney U=92, n1=n2=20, p=0.00169, one-tailed, Bonferroni correction for 12 comparisons). Median distances travelled in the rp75 and rp50 runs were also more than twice that in the default runs, with the former distributions found to be higher than the latter according to statistical significance tests (rp75 Mann-Whitney U=91, n1=n2=20, p=0.00159, one-tailed; rp50 Mann-Whitney U=99, n1=n2=20, p=0.00307, one-tailed, Bonferroni correction for 12 comparisons). For episode time (amount of timesteps before walker failed), all medians were the maximum value (1000, the walkers lasted till the end of the simulation) except for the rp25 runs, which failed to produce a long-lasting gait. Figure 5.6 includes (bottom-left) a high-fitness gait produced by the rp75 (0.75 radius-multiplier) runs, without action noise. The walker dragged itself forward with

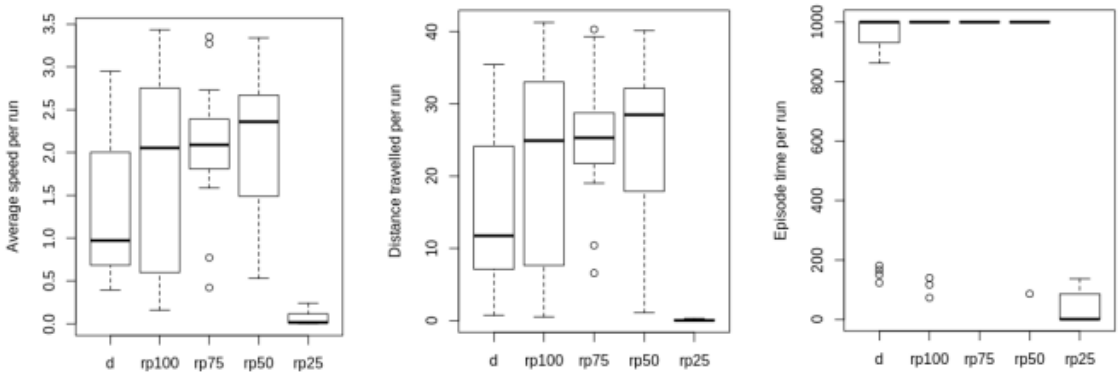

Figure 5.4: **Balance Results.** Results for the default evolved behaviours (d) and for walkers evolved with balance-circle radius-multipliers 1.00 (rp100), 0.75 (rp75), 0.5 (rp50) and 0.25 (rp25): speed (left), distance (middle) and time (right) for each set of 20 runs. The rp50 runs had the highest median speed and distance travelled. The rp75 and rp50 were found to be faster and travelling further than the default according to statistical significance tests. The rp100, rp75 and rp50 runs had the highest proportions of walkers reaching the maximum length of time allocated without failing. The rp75 runs were found to last longer than the default according to a statistical significance test.

one leg while pumping its arm, a behaviour unseen in the default gaits. This was likely due to the increased momentum provided to assist in the stability of the gait, as seen from Collins *et al* [18].

5.3.3 Combined Enhancement

Figure 5.5 shows the results for the default (d) evolved behaviours, for walkers evolved with the most successful control-cost and balance-enhancement parameters (s500 and rp75) and for those evolved with the two enhancements combined (s500 combined with rp75). As the combo was tested against the default, s500 and rp75 in three metrics (speed, distance travelled and episode time), a Bonferroni correction was applied for 9 comparisons, narrowing the significance margin to 0.0056. Median speed and median distance in the combined-enhancement were again more than twice those in the default runs, demonstrating that the two enhancements do not interfere with each other, with the former distributions found to be higher than the latter according to statistical significance tests (speed Mann-Whitney U=79, n1=n2=20, p=0.00052, one-tailed, distance Mann-Whitney U=74, n1=n2=20, p=0.00032, one-tailed, Bonferroni correction for 9 comparisons). I found no increase in median speed or median

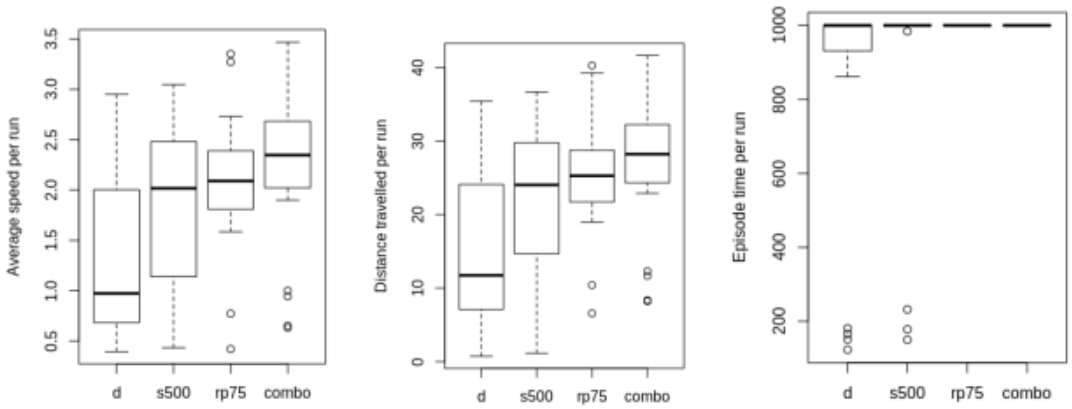

Figure 5.5: **Combined Results.** Results for the default evolved behaviours (d) and for walkers evolved with the 0.25 control-cost multiplier from 500 time-steps (s500), with the 0.75 balance-circle radius-multiplier (rp75) and with a combination of the two (combo): speed (left), distance (middle) and time (right) for each set of 20 runs. The combo had the highest median speed and distance travelled. It travelled further and faster than the default according to statistical significance tests, but was not found to be further or faster than either the s500 or the rp75. The rp75 and combo had the highest proportion of long lasting walkers.

distance between the s500 or rp75 or the combined-enhancement runs. It seems unlikely that any difference in median (between s500 and combo or between rp75 and combo) would be large even if found to be bigger according to a statistical significance test. This may be because the two enhancements work in opposite directions, in that one is a reduction in control cost, allowing greater movement, while the other is a restriction on movement. For episode time (amount of timesteps before walker failed), the combined-enhancement runs matched the rp75 runs in having all gaits reach 1000 time-steps, although it is possible that increasing the 1000-timesteps limit could reveal a difference. It should also be stated that in all three metrics, the distribution of the combo more closely resembles the that of the rp75 than the s500. This suggests that the rp75 has more impact than the s500, and potentially therefore that balance is more important than efficiency or novelty for robustness. In the absence of noise, the highest-fitness combined-enhancement gaits showed no noteworthy novelties, with all featuring either wide-legged shuffling with no leg crossover or single-leg dragging gaits. An example of the former can be seen in Figure 5.6 (bottom-right).

Figure 5.6: **Noiseless Gaits.**(Reading from left to right, then top to bottom). Gaits without noise. In a high-fitness gait from the default runs (top-left) the biped shuffled by alternating its knees in an unnatural motion. In a high-fitness gait from the s500 runs (0.25 multiplier from 500 time-steps, top-right) the biped put one leg in front of the other in succession, with a much wider range than the default's shuffling behaviours. In a high-fitness gait from the rp75 runs (bottom-left) the biped pulled itself forward on one leg and pumps one arm for momentum, something previously unseen. In a high-fitness gait from the combo runs (bottom-right) the biped shuffled similarly to the default gait, but with a wider spread of the legs. (The darkness at the end of the figures occurs as the humanoids walk out of the range of the white floor texture. The floor texture and shadows have been altered from the original task environment for clarity.) (Videos of these gaits can be found at https://github.com/benjack795/bipedal-methods.)

5.3.4 Robustness to Action Noise

Figure 5.7 shows the degradation of average speed, distance travelled and episode time with increasing levels of action noise. The combined-enhancement runs were much more robust to action noise than the default, particularly around noise=0.4, not dropping off in fitness in the presence of low noise as quickly as the other types. The rp75 and s500 runs produced intermediately robust gaits at low levels of action noise but dropped off much quicker as noise increased when compared to the combination runs.

Figure 5.8 provides a closer look at the noise results at the 0.4 level, at which the combined-enhancement runs exhibited consistently superior results. In contrast to the without-noise results, the combined-enhancement results now show large improvements over the individual control-cost and balance enhancements according to statistical significance tests. The combination runs do not drop off in fitness in the presence of low noise as quickly as the other types. As the combination was tested against the default, s500 and rp75 at 0.4 noise for three metrics (speed, distance travelled and episode time), a Bonferroni correction was applied for 9 comparisons, narrowing the significance margin to 0.0056. Median speed in the combined-enhancement runs was more than twice that in the default runs and the combined-enhancement distribution was found to be higher than the default runs (Mann-Whitney U=79, n1=n2=20, p=0.00052, one-tailed, Bonferroni correction for 9 comparisons) and the rp75 runs according to statistical significance tests (U=99, p=0.00307, one-tailed, Bonferroni correction for 9 comparisons). Median distance in the combined-enhancement runs was more than twice that in the default, s500 and rp75 runs, and the combined-enhancement distribution was found to be higher than the default runs (Mann-Whitney U=53, n1=n2=20, p=0.0003, one-tailed, Bonferroni correction for 9 comparisons), the s500 runs (U=88, p=0.00122, one-tailed, Bonferroni correction for 9 comparisons) and the rp75 runs according to statistical significance tests (U=70, p=0.00022, one-tailed, Bonferroni correction for 9 comparisons). Median episode time in the combined-enhancement runs was also more than twice that in the default, s500 and rp75 runs, and the combined-enhancement distribution was found to be higher than the default runs (Mann-Whitney U=67, n1=n2=20, p=0.00016, one-tailed, Bonferroni correction for 9 comparisons), the s500 runs (U=74, p=0.00032, one-tailed, Bonferroni correction for 9 comparisons) and the rp75 runs (U=62, p=0.00009, one-tailed, Bonferroni correction for 9 comparisons) according to statistical significance tests. Figure 5.9 shows the four previous high-fitness gaits under noise level 0.4. The

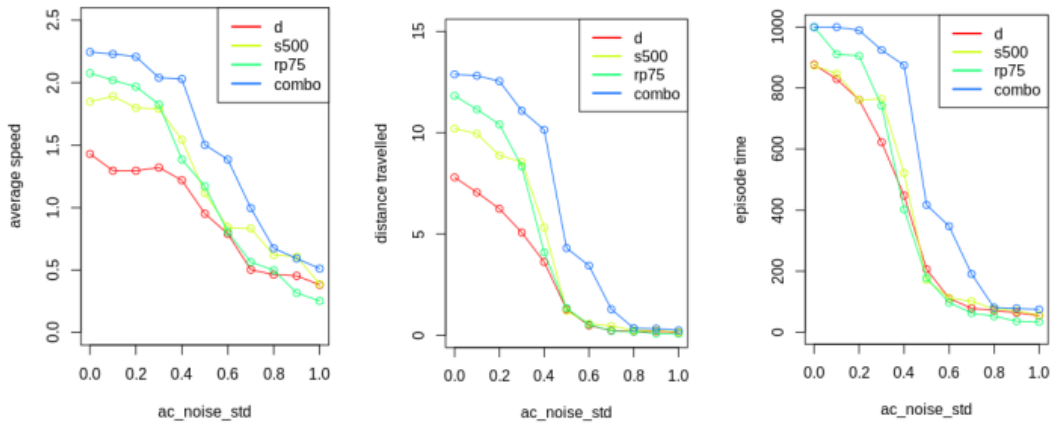

Figure 5.7: **Action Noise Scaling.** Results from scaling action noise from 0 to 1 for the default (d), 500 delay (s500), 0.75 (rp75) radial multiplier and combination of the two (combo): speed (left), distance (middle) and time (right) averaged over each set of 20 runs. The combo runs perform the best across all three results as noise is increased, eventually decreasing in fitness to that of the others, which begin to lose fitness at lower noise levels.

default (top-left), s500 (top-right, 0.25 control-cost multiplier from 500 time-steps) and combined-enhancement (bottom-right) runs produced similar gaits to Figure 5.6 but with much wider motions, flailing limbs more. The rp75 (bottom-left, 0.75 radius-multiplier) gait arches the walker's back a little more but otherwise remained more stable, continuing to pump its arm.

5.4 Contributions

The contributions made by this work are as follows:

- I reduce control cost within the fitness function, based upon an adjustment to the torque limit in the Solomon *et al* system in the previous Chapter, prioritising novelty over efficiency. When control cost was reduced to a quarter of the default, from the 500th time-step (the halfway point for a full-length episode), median speed and median distance both doubled. I also terminate walking when the torso's centre of mass moves outside a circular support polygon centered at the midpoint between the walker's feet, based around an existing centre of mass condition within the system and the importance of balance to bipedal tasks overall. When the circle's radius was 0.75 times current distance between the

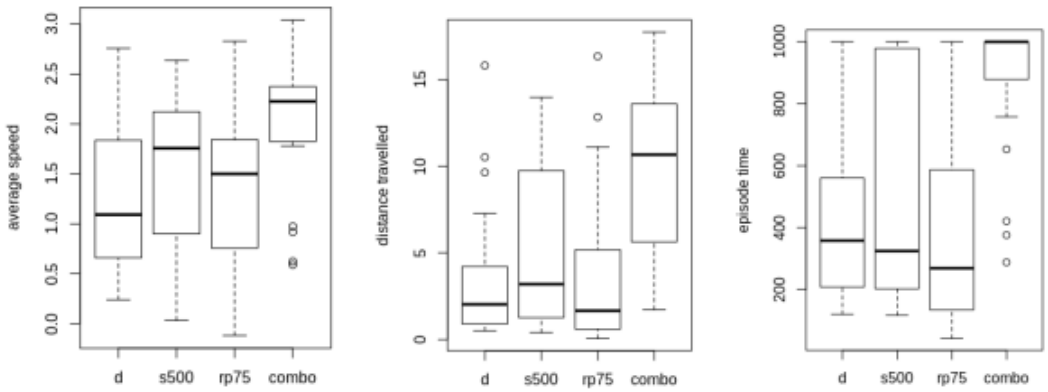

Figure 5.8: **Action Noise Results.** Results with action noise=0.4; for the default evolved behaviours (d), for walkers evolved with the 0.25 control-cost multiplier from 500 time-steps (s500), with the 0.75 balance-circle radius-multiplier (rp75), and with a combination of the two (combo): speed (left), distance (middle) and time (right) for each set of 20 runs. The combo runs have the highest median values for all three results. They were also found to be faster, travel further and last longer than all the other variations according to statistical significance tests.

midpoint and (either) foot, median speed and median distance are both found to be higher according to a statistical significance test. This demonstrates robustness of these methods in a new system. This matches my Chapter aim to test the robustness of my methods and my general aim to increase walker fitness.

- I produce notable gaits using the enhancements with a more pronounced stance and swing phase. One puts one leg in front of the other in a clearer fashion than any gait produced using the default fitness function, getting around the issue of shuffling behaviours. The other produces an arm pumping motion similar to arm swinging in humans. This demonstrates my results from these techniques approaches those produced by machine learning. This matches my Chapter and general aim to produce complex behaviour.

- I evaluated evolved walkers with the addition of noise to their actions. The combination runs do not drop off in fitness in the presence of low noise as quickly as the other types. In contrast to the without-noise results, the combined-enhancement gaits showed large improvements over those from the individual control-cost and balance enhancements, with median speed, distance and time

Figure 5.9: **Action Noise Gaits.** (Reading from left to right, then top to bottom). Gaits with noise = 0.4, from the neural network controllers shown in Figure 5.7. In the default-runs gait (top-left) the biped shuffled by alternating its knees in a more erratic way than before. In the s500-runs gait (top-right) the biped still put one leg in front of the other, but much more loosely. In the rp75-runs gait (bottom-left) the biped used one arm for momentum and was not affected too heavily by the noise, only bending its back more. In the combo-runs gait (bottom-right) the biped shuffled on its knees with a wide stance, making more flailing motions than previously. (The darkness at the end of the figures occurs as the humanoids walk out of the range of the white floor texture. The floor texture and shadows have been altered from the original task environment for clarity.) (Videos of these gaits can be found at https://github.com/benjack795/bipedal-methods.)

higher than the default and individual-enhancement gaits at intermediate levels of noise according to a statistical significance test. This demonstrates the combined enhancement drops off slower in the presence of low noise and matches my general aim to increase walker fitness. The distribution of the combo runs also more closely resembles the that of the rp75 than the s500. This suggests that the rp75 has more impact than the s500, and potentially therefore that balance is more important than efficiency or novelty for robustness.

5.5 Discussion

An interpretation of my results seen here and also in the last Chapter is the importance of fitness functions when enhancing an evolutionary system. I modified the fitness function directly for my first enhancement in this work, applying a scalar multiplier to one of the terms (control cost) after a given time period. This resulted in increased median fitness. I also introduced a new failure condition to the task, a support polygon based constraint for walker centre of mass. This also resulted in increased fitness according to a statistical significance test, with the combination runs also resembling this distribution. This can be interpreted as further evidence that fitness functions are the most important aspect to consider when enhancing an evolutionary system. This interpretation could again help focus the methodology of further evolutionary works towards complex behaviour by encouraging focus on fitness function enhancements. It could also demonstrate the enhancement philosophy is robust here, due to a second success across a more modern system with a different neural network controller type and evolutionary process.

Another interpretation of my results seen here and also in the last Chapter is the the idea of sacrificing efficiency for novelty. By reducing the importance of the control cost term in the fitness function, I allowed for more walker motion at the cost of less efficiency. This produced a successful walking gait that put one foot in front of the other instead of shuffling, something previously unseen in default runs of the system. This can be interpreted as further evidence that prioritising novelty over efficiency can lead to more complex behaviour in evolved agents in other works. Once again this could lead to other works featuring more complex evolved behaviour in their agents, furthering our understanding of the evolution of complex behaviour in both agent-based systems and the biosphere. It could also further demonstrate the robustness of the philosophy, similarly to the fitness function philosophy above, due

to a second success across a more modern system with a different neural network controller type and evolutionary process.

The results in this work can also be interpreted as justification for the rationale of choosing evolutionary systems over other types of learning system. In this work I produce results that are found to be fitter than the original system across two separate enhancements according to statistical significance tests, which was already capable of matching reinforcement learning methods on the humanoid walking task. This could be interpreted as further evidence of evolutionary systems being a potential alternative to reinforcement learning on more complex tasks. This could lead to other works choosing to use evolutionary systems, giving them the benefits of biological insight into where each behaviour has originated from in evolutionary space, and the ability to compare that to the biosphere. This could also lead to further works benefiting from incremental evolution, the other main benefit of evolutionary systems. By breaking a task down into sub-tasks and using an evolutionary system to evolve between them, a complex task can be greatly simplified without having to worry about over-fitting to training data.

A final interpretation of my results here is the precise nature of the interaction between the two enhancements used in this work. The two enhancements can be seen as contrasting in nature: whilst the control cost enhancement aims to encourage more motion by reducing efficiency, the balance enhancement restricts motion to improve robustness and stability. Whilst the results from either enhancement individually improved fitness, when combined together they performed better in the presence of noise as well, according to statistical significance tests. This can be interpreted as a requirement for robustness: whilst prioritising novelty over efficiency can lead to more complex behaviour, combining both a restrictive and a relaxing enhancement to a fitness function can result in greater robustness instead. However, the distribution of the combination runs resembled the rp75 runs much more than that of the s500, suggesting the ideal balance may not be an equal weighting. This philosophy could lead to considerably improved behaviour in works involving bipedal tasks, as robustness is essential to maintain agent balance throughout behaviour. This could also lead to weighting metrics from the work of Conti *et al* for fitness against novelty having further application balancing encouragement enhancements and restriction enhancements for increased robustness [20].

5.5.1 Acknowledgements

Our code is based on code from Such *et al* [81], which was in turn based on code from Salimans *et al* [69]. I thank these researchers for making their code available. My code is also available via GitHub at *https://github.com/benjack795/bipedal-methods* for exact specification and future reproductions.

Chapter 6

A Simple 3D-Only Evolutionary Bipedal System with Albatross Morphology for Increased Performance

6.1 Chapter Aims

- In this Chapter I aim to utilise a baby albatross-based morphology with shorter legs and a spherical body and a square base to allow bipedal agents to tuck in their legs and roll back onto the flat base, after toppling. This morphology is based on the support polygon philosophy from the previous Chapter and focuses on enforced balance to produce improved fitness in the Solomon *et al* system. This is in line with my aim to increase walker fitness and also my aim to use morphology to simplify the bipedal walking task.

- In this Chapter I also aim to utilise a baby albatross morphology featuring shorter legs and a vastly larger sphere attachment, resulting in a rounded but close-to-flat lower body without a square base. This morphology will also feature modified the knees and feet, bending inwards to fold away, and allowing agents to lift themselves off the ground like a scissor lift. This morphology is based on the control cost penalty in the previous Chapter. The supporting large sphere allows for more motion, aiding walker balance to produce improved fitness in the Solomon *et al* system. This is in line with my aim to increase walker fitness and also my aim to use morphology to simplify the bipedal walking task.

- I also aim to test incrementally evolving agent morphology simultaneously via a separate morphological genotype, encoding the leg length proportions of the

albatross walkers. Starting from both incremental high-fitness and randomly initialised positions, I aim to return walker leg proportions back to the default whilst retaining any fitness benefits achieved by the albatross proportions. This is in line with my aim to use morphology to simplify the bipedal walking task.

6.2 Methodology

6.2.1 Rationale

In this Chapter I utilise morphology in order to improve fitness and produce more successful walker gaits. The reasoning for this decision is twofold. Firstly, morphology is often shown to be used in the biosphere - the most relevant example being the albatross feathers seen in this paper, making it a natural solution to to the bipedal walking problem. Feathers have been demonstrated to assist with complex tasks by both Shim *et al* and Chang *et al* [72] [15]. Secondly, morphology allows for a less complex neural network controller, by replacing computation for aspects of a task. This is particularly useful in an already complex task, such as bipedal walking, which requires constant balance. This can be seen from McGeer's passive dynamic walker, using its rounded feet to maintain balance [51]. The *square-base* morphology was chosen in order to allow walkers to self-right with the sphere and re-align themselves with the square base. It was conceived as a physical version of the support polygon fail condition from the previous Chapter, with a polygon formed around the legs keeping the walker balanced. The *big-tucked* morphology was chosen in order to allow walkers to roll upright on a much larger sphere, and then extend the tucked legs to re-align themselves instead, like a scissor lift. It was conceived from the rationale of allowing novelty in motion from the previous Chapter, this time through morphology instead of a fitness function. An alternative could have been to introduce soft materials, seen from Lipson *et al*, producing soft body locomotion in voxel-based creatures [48]. However, the ODE engine does not support flexible bodies. Another alternative may have been to use an upper body with swinging arms, mimicking the albatross' wings in order to stabilise motion, seen in human arm-swinging by Collins *et al* [18]. However, this would have complicated the model heavily, requiring more actuators and therefore a more complex network.

After the previous Chapter demonstrated the success of fitness function enhancements on the Salimans *et al* Mujoco system, for this Chapter I decided to return to the Solomon *et al* ODE based system. Similarly to the previous Chapter, to avoid

feelings of a disconnect between Chapters - I again clarify the rationale behind this decision. Firstly, the Mujoco system featured a large and complex deep network, running across a master-worker multi-threaded implementation, designed to be run on high end hardware, compared to the relatively simple neural network controller and implementation of the Solomon *et al* system. Modifying morphology and joint angles in the Mujoco system adds several new inputs and outputs to the neural network controller, which would result in thousands of new parameters in the fully connected deep network, increasing runtime. For this reason I chose to experiment on the Solomon *et al* system first. Secondly, in this Chapter I attempted to investigate incremental morphological evolution. As the code produced by Such *et al* is not my own, it would have taken a considerable length of time to rework in order to enable this functionality in the system, particularly given the scope of the parameters involved. Finally, the simpler neural network controller in the Solomon *et al* system is more sensitive to morphological changes. Changes in morphology are harder to attribute to a more complex neural network controller such as the one featured in the Salimans *et al* system. The complex neural network controller can adapt around the changes in morphology, instead of learning ways to utilise the morphology to replace computation, which is an aim in this thesis. An alternative to the ODE engine would be to work the morphological enhancements into smaller changes in the Mujoco engine, to avoid the time constraint issues. However, smaller changes would be harder to attribute clear success and complexity to, compared to more clear, direct morphological influence.

6.2.2 Previous System

In Chapter 4 I produced an enhanced version of the Solomon *et al* system, which this work is built upon. I chose the their work for re-implementation as their rough terrain results demonstrated robustness of behaviour, using a deliberately simple neural network controller. I utilised four main changes in order to evolve walkers into a 3D physics engine from scratch with the most basic control scheme. Further minor re-implementation changes are listed in Chapter 4. The major changes were the removal of a control cost limit after a given time period, allowing a flight phase, replacing the network PD controls with ODE engine parameter methods and retaining a larger proportion of agents for elitism. These allowed the agents to exhibit more motion rather than prioritising efficiency. By applying these modifications to the system, I was able to evolve successful bipedal agents in the 3D physics engine with no prior bootstrapping, despite being constrained to 2D motion. I created my re-implementation with

the Open Dynamics Engine [76]; it featured bipedal agents configured with identical body parameters to the original work, seen in Figure 6.1. The genetic algorithm used was identical, with a population size of 150 for 500 generations. My winner was not selected from the elites of several final generations; instead the fittest of the final generation was chosen, as stochastic rough terrain was not tested. Fitness was changed back to distance travelled forwards, instead of steps, for this Chapter, as this was the metric used in the original paper. Torque limits were also disabled for the entirety of agent lifetimes, instead of after a given time period, due to the different mass of the albatross morphologies resulting in different torque usages. There were also no lateral joints or inputs in my 3D model, unlike in Chapter 4, to reduce the size of the search space. There were also no point mass blocks on the thighs and shanks, and all masses were initialised in the same way. My agents contained six linear reactive controller neural networks, with inputs configured identically to the Local Proportion neural network controllers described above. I initialised agents standing on one leg with the other swinging upwards, as in the original, with a small upward force of 0.25 units. By starting in a swinging motion, the walkers could transition into a cyclic stepping motion with less difficulty.

6.2.3 Albatross Morphology

The baby albatross, seen in Figure 6.2, has a set of fluffy feathers on its underbelly that soften impacts and allow it to roll upright to regain balance. Feathers are theorised to have emerged from scales in order to repel water and produce an airtight barrier [26]. Their softness was also proven to assist with adjusting flight trajectory, by both simulated [72] and real world agents [15]. After the success of enhancements on both the Solomon *et al* and Salimans *et al* systems, I will achieve my third objective, utilising morphology alongside control methodologies, on the Solomon system by applying albatross-based morphologies to my system from Chapter 4.

6.2.3.1 Square-Base

I refer to the first albatross morphology tested as the *square-base* morphology. The morphology featured a large sphere segment attached to the upper body, with a lower cuboid segment attached to the sphere to give the sphere a square base. This would resemble the lower body of a baby albatross. The legs were also scaled to half length to allow the sphere to contact the ground. Shown in Figure 6.3, this morphology would allow bipedal agents to tuck in their now-shorter legs into the sphere and roll

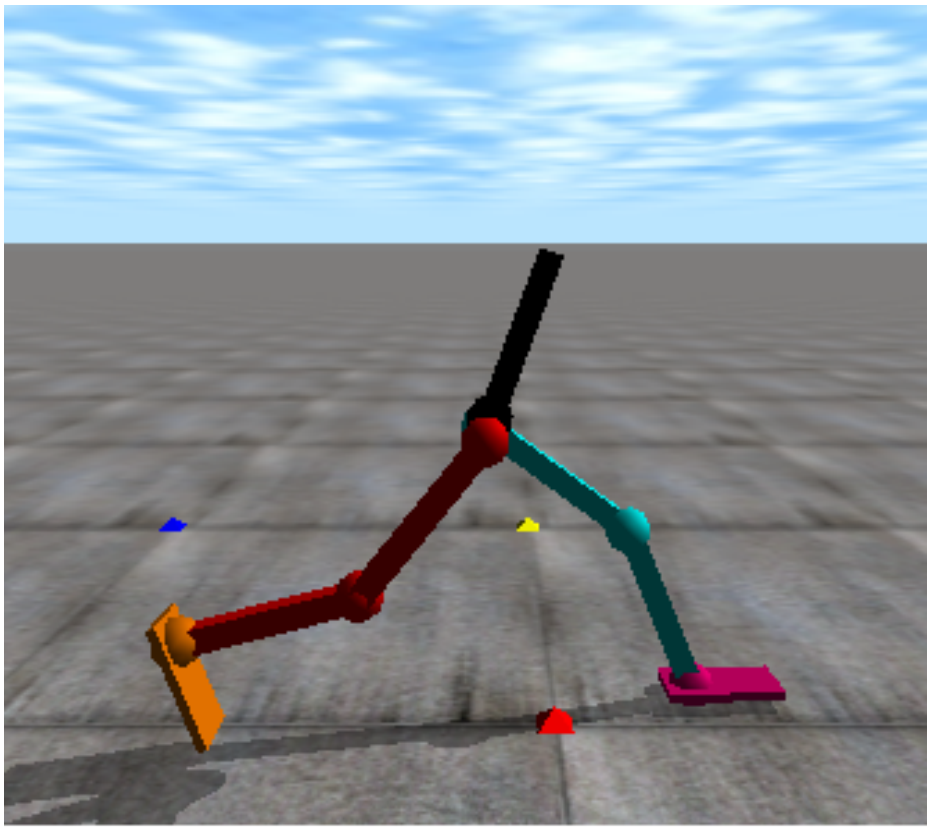

Figure 6.1: **My previously enhanced walker in ODE.** My re-implemented system mostly matches the design of the original, with identical joints and morphological parameters. The main differences to the original involved a reduction of control cost in the fitness function, increased elitism, a flight phase and removal of PD Controls. The remaining minor re-implementation changes are documented in Chapter 4. It steps forward on flat terrain.

Figure 6.2: **The Baby Albatross** The baby albatross is seen here with its soft feathered underbelly which it uses to cushion falls and roll upright [49].

back onto the flat base, after toppling. Rolling onto the square base would allow them to realign themselves using its flat edges, keeping the legs parallel with direction of travel. In the previous Chapter, I utilised a support polygon based methodology to enforce balance on walkers. I terminated walker agents if their centre of mass' x or y position travelled outside a circular range drawn on the floor around their feet. The polygon used the position of each foot as opposite points of the circle's diameter. This forced the walker to prioritise balance. The *square-base* morphology is based on this philosophy, forcing the walker to roll back into place and remain balanced. To achieve this, the sphere and cuboid were exempt from the system's fall condition when they contacted the ground. The original fall condition stated that when a component of the walker that was not the calf or foot contacted the ground, the walker was terminated for falling. In order to use the sphere as a rolling support, the sphere attachment had to be exempt from this fall condition. As the sphere was larger than the majority of the legs and the entire upper body, this means that the morphology was now unable to fail this condition. Whilst this does provide a considerable advantage, it does not guarantee higher fitness or longevity - an agent could flail endlessly without moving forwards despite being unable to fall over. This albatross morphology was tested in full 3D.

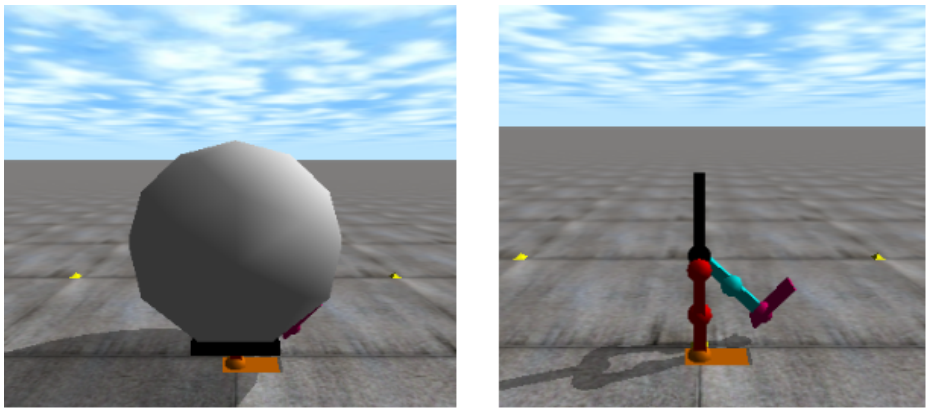

Figure 6.3: **The *square-base* morphology.** (left) The *square-base* morphology with the square-based sphere visible. (right) The same walker without, demonstrating the shorter legs. The sphere will help it to roll back onto the base and re-align itself when it tilts too far over.

6.2.3.2 Big-Tucked

I refer to the second morphology tested as the *big-tucked* morphology. Shown in Figure 6.4, it featured a vastly larger sphere attachment, resulting in a rounded but close-to-flat lower body without a square base. A small spherical counterweight was also added inside the larger sphere. These would enable agents to roll back upright, much like the *square-base* morphology. The legs were also scaled to half length as in the *square-base* morphology. The main aspect of this morphology, however, was the knees and feet. The baby albatross exhibits legs that appear to bend inwards, to fold away when they sit; in fact this is an ankle joint, with an elongated foot. I inverted my agents' knee joint limits to resemble this. Aiming for this to allow agents to similarly tuck their legs away into the larger sphere, these would allow them to lift themselves off the ground, akin to a scissor lift. I shortened the feet to prevent the agents from using the tucked legs to drag themselves along instead of stepping. In the previous Chapter, I utilised a reduction on the control cost penalty in the Salimans *et al* system's fitness function, in order to allow for more motion in evolved gaits and therefore more complex behaviour. The *big-tucked* morphology is also based on this philosophy, encouraging walkers to produce more advanced motion via a supporting large sphere, aiding their balance and allowing for larger movements without failure. This should then exhibit more motion than the default, leading to more life-like gaits beyond basic compass steps and shuffles. This morphology was also unable to fall due to the sphere support not being included in the fall condition by design, and suffers the same technicality mentioned in the *square-base* section above. This albatross

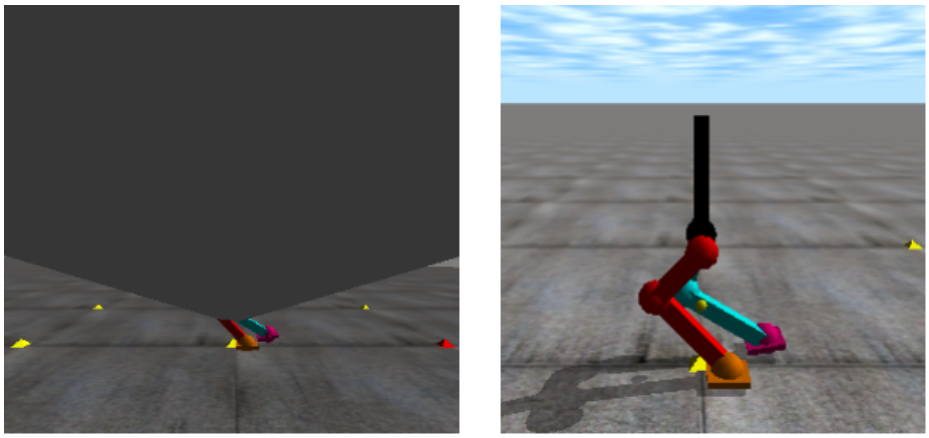

Figure 6.4: **The *big-tucked* morphology.** (left) The *big-tucked* morphology with the larger sphere visible. (right) The same walker without, highlighting the inverse knees, shortened feet and white counterweight between the legs. The big sphere will help it to roll back upright onto the tucked legs, which it can then use to lift itself back up into a walking gait.

morphology was also tested in full 3D.

6.2.3.3 Incremental Morphological Evolution

I also tested evolving the two albatross morphologies. I created a separate genotype for the agents' morphology. This featured two genes: a single multiplier value, applied to both the upper and lower leg lengths, and a mutation rate. I limited the multiplier gene value to being between the albatross leg length (multiplier value of 1) and the default length (multiplier value of 2). In this way, walkers would be helped to return to default leg proportions with any improvements from the albatross leg proportions intact. This body genotype was simultaneously mutated and crossed over with the same methodology as the neural network genotype. As mentioned above, the shorter-legged albatross morphologies should evolve back to default proportions, whilst retaining their increased fitness. I tested incrementally evolving high-fitness agents from the two albatross morphologies with preloaded neural network and morphology genotypes. I then tested agents with both genotypes randomly initialised alongside these.

6.3 Results

6.3.1 Base Results

I evolved each tested morphology in the previous enhanced system as a set of twenty runs, each with a population of 150 over a period of 500 generations, chosen as the largest number of generations for a large enough population that could be executed within a reasonable runtime. Twenty was chosen as the generally accepted minimum sub-group sample size for quantitative research. Unlike Chapter 4, I measured fitness as distance travelled forwards from the origin instead of steps, to better match the original work of Solomon *et al*. Agents failed when they exceeded a 60 second lifetime. This was the only termination condition for the walkers due to their inability to be affected by the systems fall condition. The default morphology walkers here were rerun without a torque limit, unlike the temporary torque limit from Chapter 4, for a fair comparison with the new morphologies also not using a torque limit. Compared to the original work of Solomon *et al*, my default morphology walkers here achieved similar fitness in 3D, although without an efficiency constraint. As seen in Figure 6.9, they were able to achieve walking gaits in 3D, and travelled for some distance before falling over. Unlike the work of Solomon *et al*, however, I evolved all the gaits in this work in a 3D physics engine from scratch with no prior processing. I chose the work of Solomon *et al* for re-implementation as the rough terrain results demonstrate robustness of behaviour. Their walkers also feature a simpler neural network controller style than other works. In order to demonstrate notable results in this chapter, I utilise the Mann-Whitney U test for significance. This was chosen as the tests all featured the effects of a single change to the system on one continuous outcome value. As there were several tests performed in multiple groups, the Bonferroni correction was applied as a simple way to ensure there was little room for statistical error within the tests. As fitness comparisons in this section are between 6 different perturbations and the default, as well as an additional comparison between the incremental and randomly initialised morphological perturbations of each of the two morphologies, a Bonferroni correction for 8 comparisons was applied to significance tests, narrowing the significance margin down to 0.0063.

6.3.2 Square-Base and Big-Tucked

Figure 6.5 shows the fitness of the twenty winning default, *square-base* and *big-tucked* morphologies. The *square-base* morphology had a much higher median than the default, but a larger range. The *big-tucked* morphology had a slightly larger median

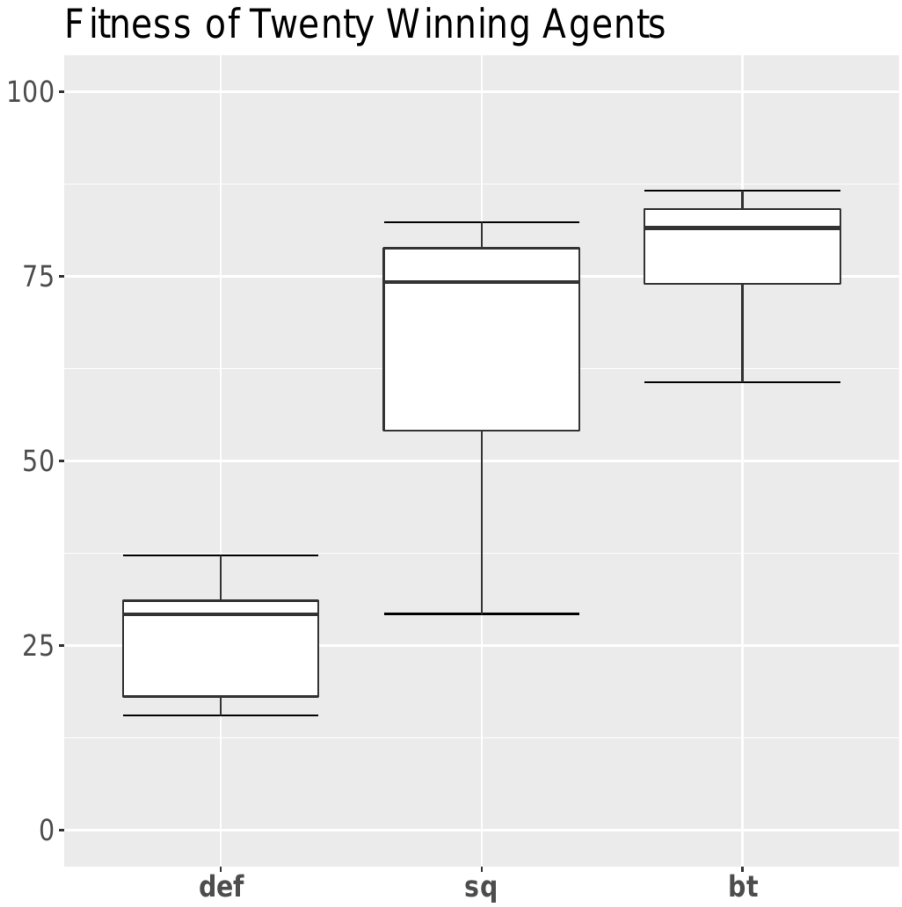

Figure 6.5: **Albatross Morphologies fitness plot.** The fitness of twenty winning agents for each set of morphologies. Def represents the default morphology, sq represents the *square-base* morphology and bt represents the *big-tucked* morphology. The *big-tucked* runs had the highest median fitness value. The *big-tucked* runs and the *square-base* runs were found to be fitter than the default runs according to statistical significance tests.

than the *square-base*, and a shorter range. Both the *square-base* (Mann-Whitney U=10, n1=n2=20, p=1.008e-09, one-tailed, Bonferroni correction for 8 comparisons) and the *big-tucked* (Mann-Whitney U=0, n1=n2=20, p=7.254e-12, one-tailed, Bonferroni correction for 8 comparisons) were found to be fitter than the default according to statistical significance tests. Figure 6.6 shows the average fitness curves for twenty of each morphology. The *square-base* and *big-tucked* morphologies evolved at a slower pace but achieved final fitness values around three times higher than the default.

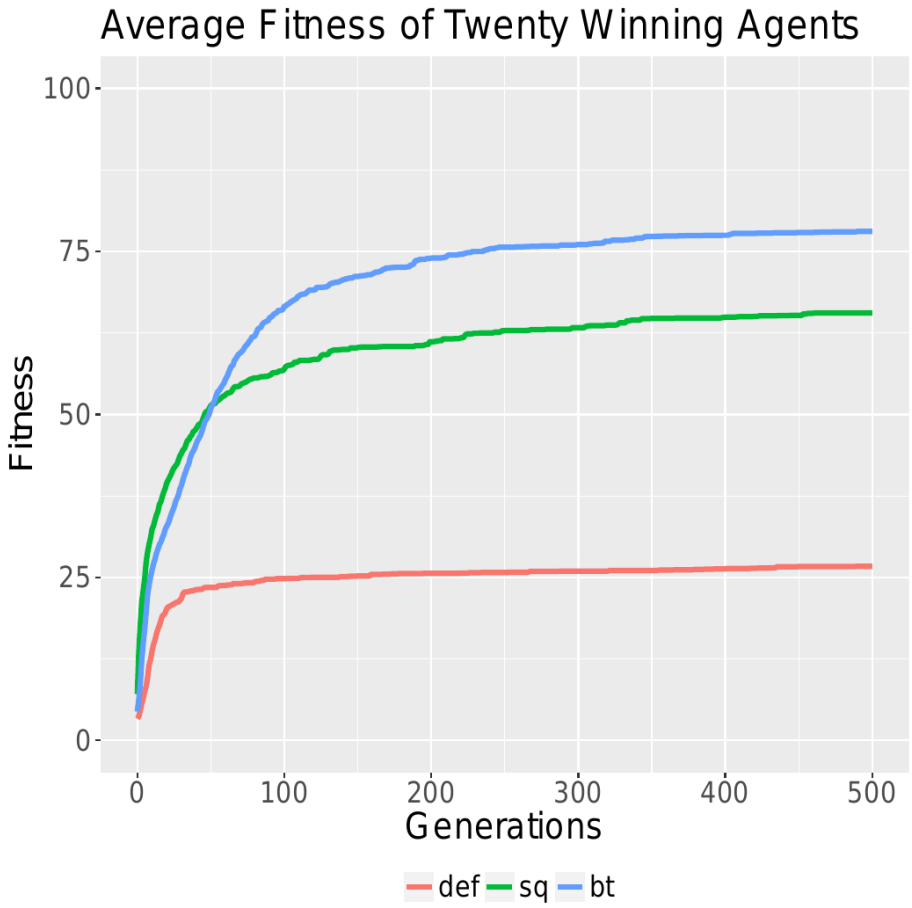

Figure 6.6: **Albatross Morphologies fitness curve.** The average fitness curves of twenty winning agents for each set of morphologies. Def represents the default morphology, sq represents the *square-base* morphology and bt represents the *big-tucked* morphology. The *big-tucked* and *square-base* runs are both fitter than the default, taking slightly longer to reach their plateaus.

6.3.3 Morphological Square-Base and Big-Tucked

Figure 6.7 shows the fitness of twenty winning agents for the default and each evolved morphology type. These included *square-base* preloaded, *square-base* randomly initialised, *big-tucked* preloaded and *big-tucked* randomly initialised morphologies. The *square-base* preloaded (Mann Whitney U=0, n1=n2=20, p=7.254e-12, one-tailed, Bonferroni correction for 8 comparisons), *square-base* randomly initialised (Mann-Whitney U=0, n1=n2=20, p=7.254e-12, one-tailed, Bonferroni correction for 8 comparisons), *big-tucked* preloaded (Mann-Whitney U=36, n1=n2=20, p=6.835e-07, one-tailed, Bonferroni correction for 8 comparisons), and *big-tucked* randomly initialised (Mann-Whitney U=0, n1=n2=20, p=7.254e-12, one-tailed, Bonferroni correction for 8 comparisons) were all found to be fitter than the default according to statistical significance tests. The *square-base* randomly initialised had a higher median, but also a larger range than the preloaded version. The *square-base* randomly initialised morphology was also found to be fitter than the preloaded version according to a statistical significance test (Mann-Whitney U=14, n1=n2=20, p=3.685e-09, one-tailed, Bonferroni correction for 8 comparisons). The *big-tucked* morphologies had similar range but drastically different medians; the randomly initialised morphology was found to be fitter according to a statistical significance test (Mann-Whitney U=0, n1=n2=20, p=7.254e-12, one-tailed, Bonferroni correction for 8 comparisons). The *big-tucked* randomly initialised median was more than three times that of the preloaded version, and more than four times the median of the default version.

Figure 6.8 shows the average fitness curves for these morphologies. All the morphologies evolved at a similar pace, with the *big-tucked*'s evolving slightly faster than the *square-base*'s. All finished above the default, with the randomly initialised versions having higher fitness than the preloaded versions on both morphology types. The difference in fitness between the two *big-tucked* morphologies was much greater than that of the two *square-base*'s. Finally, I recorded gait snapshots of high-fitness agents from the default and both randomly initialised morphologies. I took each gait snapshot over the same time period (about five seconds) at the same point during each agent's lifetime. Figure 6.9 shows a gait snapshot from the default morphology. It exhibited a stepping behaviour, but it was slow and unstable, eventually falling. Figure 6.10 shows a gait snapshot from the *square-base* randomly initialised morphology. It travelled forwards quickly with a cancan-like motion, eventually running out of time. Its body had not evolved away from that of the standard *square-base* morphology, remaining the same. Figure 6.11 shows a gait snapshot from the *big-tucked*

randomly initialised morphology. It travelled forwards at a much higher speed than the default, eventually running out of time. It evolved much longer legs than the standard *big-tucked* morphology; these were closer to the default in length.

6.3.4 Genotype Analysis

To assess which morphologies were exhibiting the most human-like gaits, I analysed their neural network genotypes. Seen in 6.12, I modelled my agents' neural network genotype as a set of network weights, a bias and a mutation rate for each of their six separate linear reactive controller neural networks. The neural network controllers produced desired angle values (seen in 6.13) for the upper body, inter-leg, stance knee, swing knee, stance ankle and swing ankle angles, which were then applied as velocity parameters to the actuators needed to change the desired angle (more details on this in Chapter 4). The stance leg was the leg balanced upon whilst the swing leg swung forwards during a gait. These roles then alternated as agents took steps. I used the simplest network configuration from the original work, Local Proportion. This fed in only the angle/derivative inputs that matched the assigned desired angle of each controller. I calculated the mean of the absolute values of each input weight in the netowork genotype, from the twenty winners for each morphology type. This allowed me to observe which morphologies' mean input weights had the higher values, and therefore how much each morphology utilised each neural network controller. Figure 6.14 shows the mean input weight values for the default, *square-base* randomly initialised and *big-tucked* randomly initialised morphologies. The *square-base* and *big-tucked* randomly initialised morphologies evolved agents with higher upper body input weights than the default. This aligned with the *square-base* and *big-tucked* randomly initialised gaits balancing for longer than the default gait. I also compared the morphologies' input weights directly without averaging. As there were comparisons between each of the *square-base* and *big-tucked* randomly initialised genotypes against the default for four inputs (upper body angle, upper body derivative, swing knee angle, swing knee derivative), a Bonferroni correction was applied here for 8 comparisons, narrowing the significance margin down to 0.0063. This demonstrated the *square-base* randomly initialised upper body angle (Mann-Whitney U=1, n1=n2=20, p=1.451e-11, one-tailed, Bonferroni correction for 8 comparisons) and upper body derivative (Mann-Whitney U=0, n1=n2=20, p=7.254e-12, one-tailed, Bonferroni correction for 8 comparisons) were found to be higher than the default

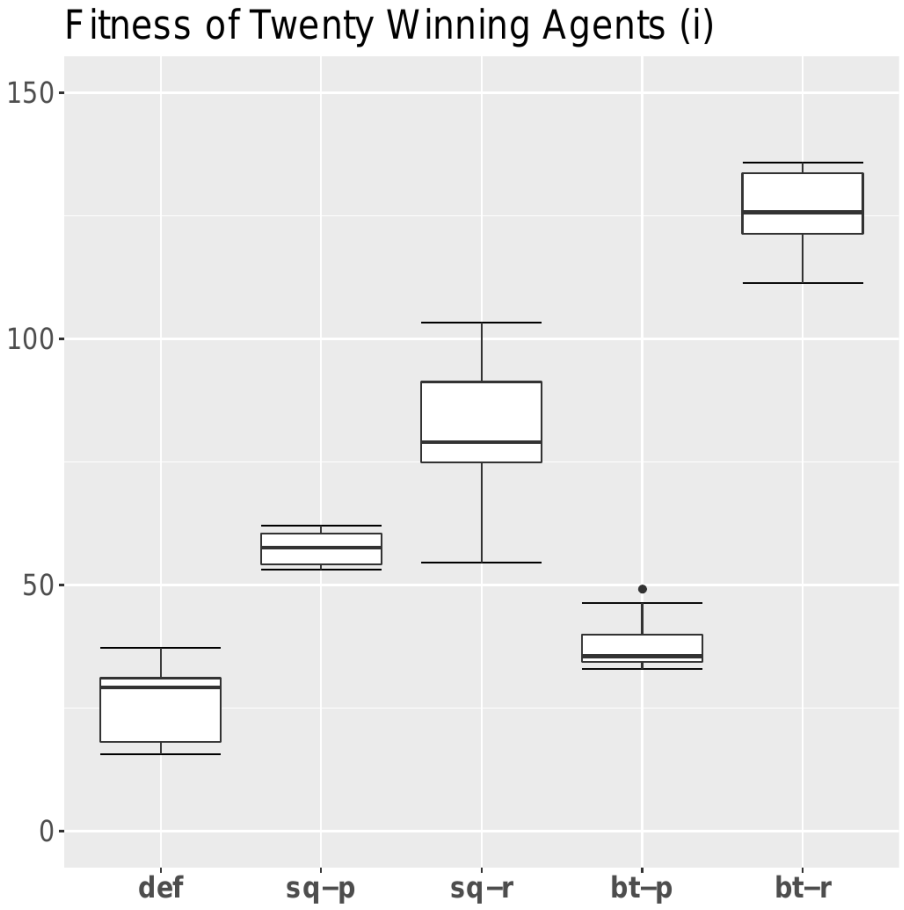

Figure 6.7: **Evolved Morphology Agents Fitness Plot.** The fitness of twenty winners for each of the evolved morphology types and the default. Def represents the default, sq represents the *square-base* morphology and bt represents the *big-tucked* morphology. A suffix of p represents a preloaded morphology. A suffix of r represents a randomly initialised morphology. The *big-tucked* randomly initialised runs had the highest medium fitness, three to four times that of the default. All variations were found to be fitter than the default, and both random variations were found to be fitter than their preloaded counterparts according to statistical significance tests.

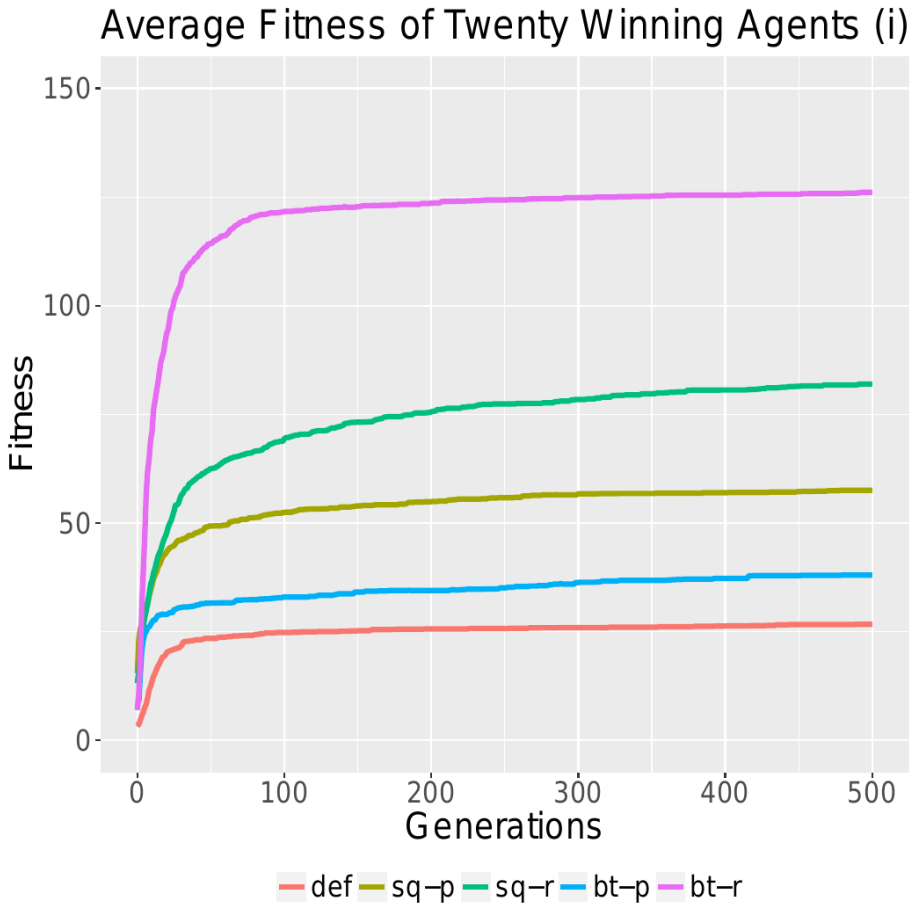

Figure 6.8: **Evolved Morphology Agents Fitness Curve.** The average fitness curves of twenty winners for each of the evolved morphology types and the default. Def represents the default, sq represents the *square-base* morphology and bt represents the *big-tucked* morphology. A suffix of p represents a preloaded morphology. A suffix of r represents a randomly initialised morphology. All the new variations scored higher fitness than the default, and the random variations featured higher fitness than their preloaded counterparts. All variations (including the default) evolved at roughly the same pace.

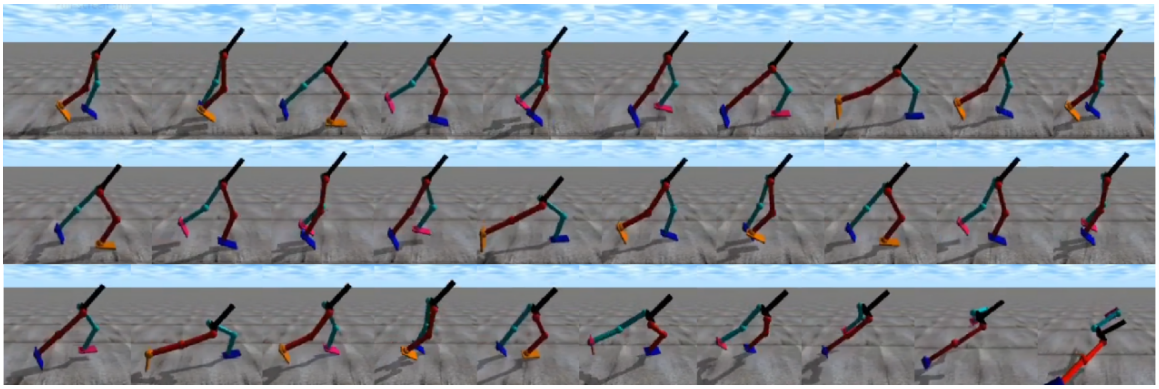

Figure 6.9: **Default Morphology Gait Example.** (left to right, top to bottom) A high fitness gait from the default morphology. It steps forward with one foot at a time, although its upper body is leaning forwards. This eventually results in a fall halfway through a step.

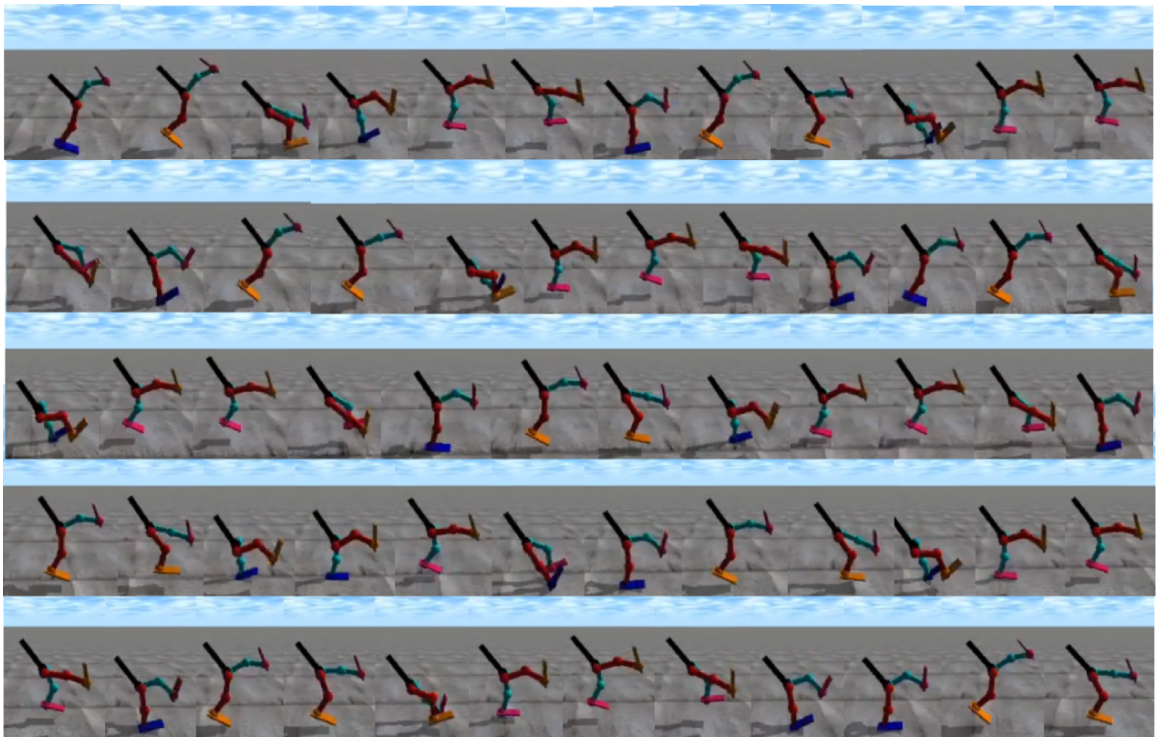

Figure 6.10: ***square-base* Morphology Gait Example.** (left to right, top to bottom) A high-fitness gait from the *square-base* randomly initialised morphology. The square based sphere attachment is not drawn. It travels forward with a cancan-like motion, keeping both legs half extended. It reaches the maximum allowed time for episodes. Its morphology is short legged, similar to the original albatross morphologies.

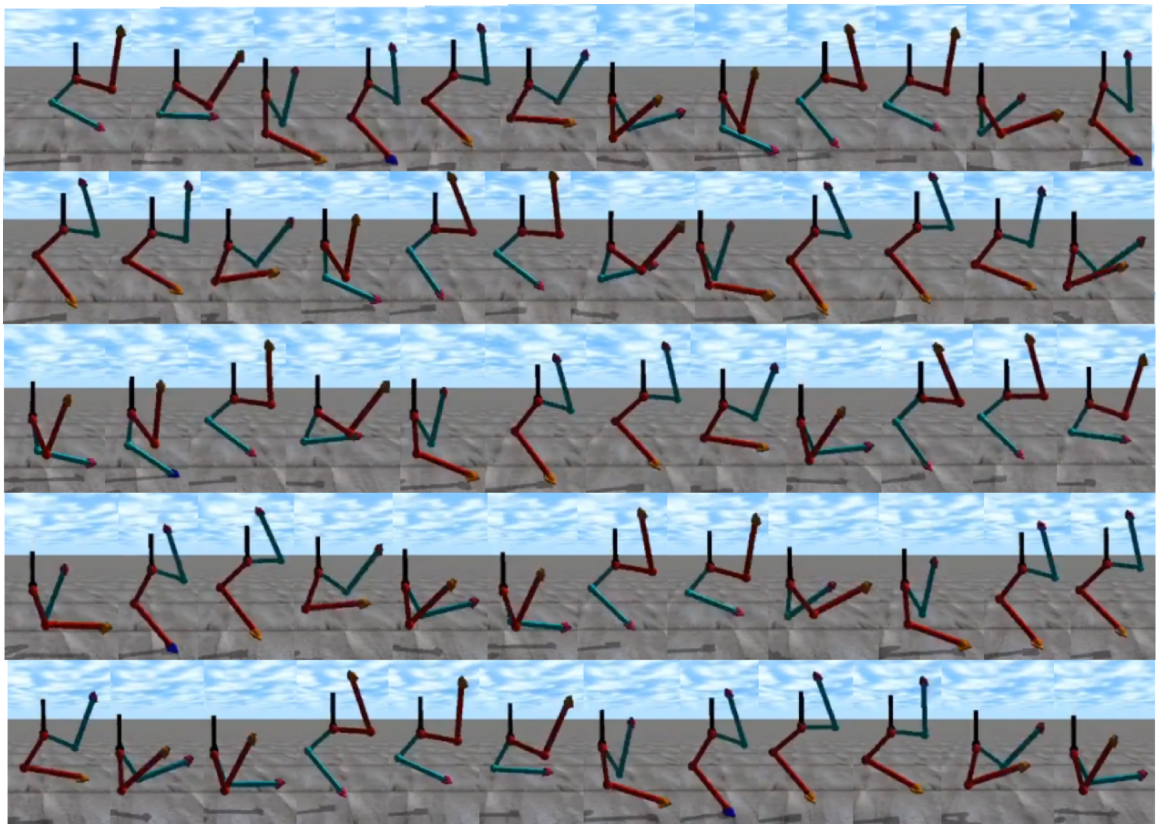

Figure 6.11: ***big-tucked* Morphology Gait Example.** (left to right, top to bottom) A high-fitness gait from the *big-tucked* randomly initialised morphology. The large sphere attachment and counterweight are not drawn. It extends its knees as it puts its legs forward, jumping along at a very fast pace and eventually running out of time. It has evolved longer legs, allowing it to take bigger steps.

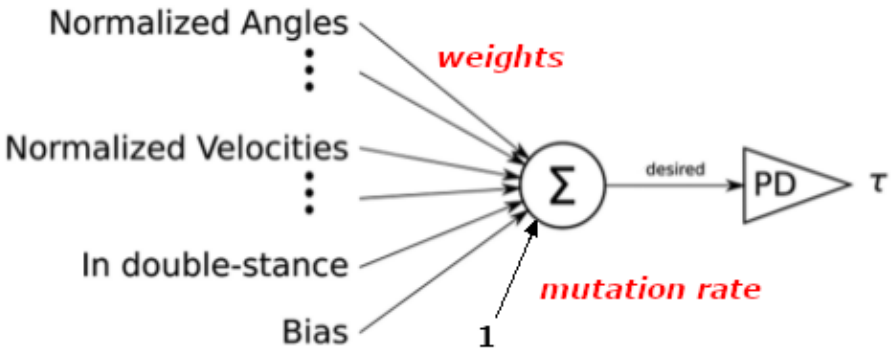

Figure 6.12: **The network topology and biped designed by Solomon *et al.*** (left) Their networks featured a simple summation of inputs, multiplied by their respective evolved weights, alongside an evolved mutation rate, to produce a torque value for each desired angle. (right) The model produced by Solomon *et al* can be seen with labelled lateral joints walking on rough terrain [77]. It also featured six sagittal joints at the hips, knees and ankles.

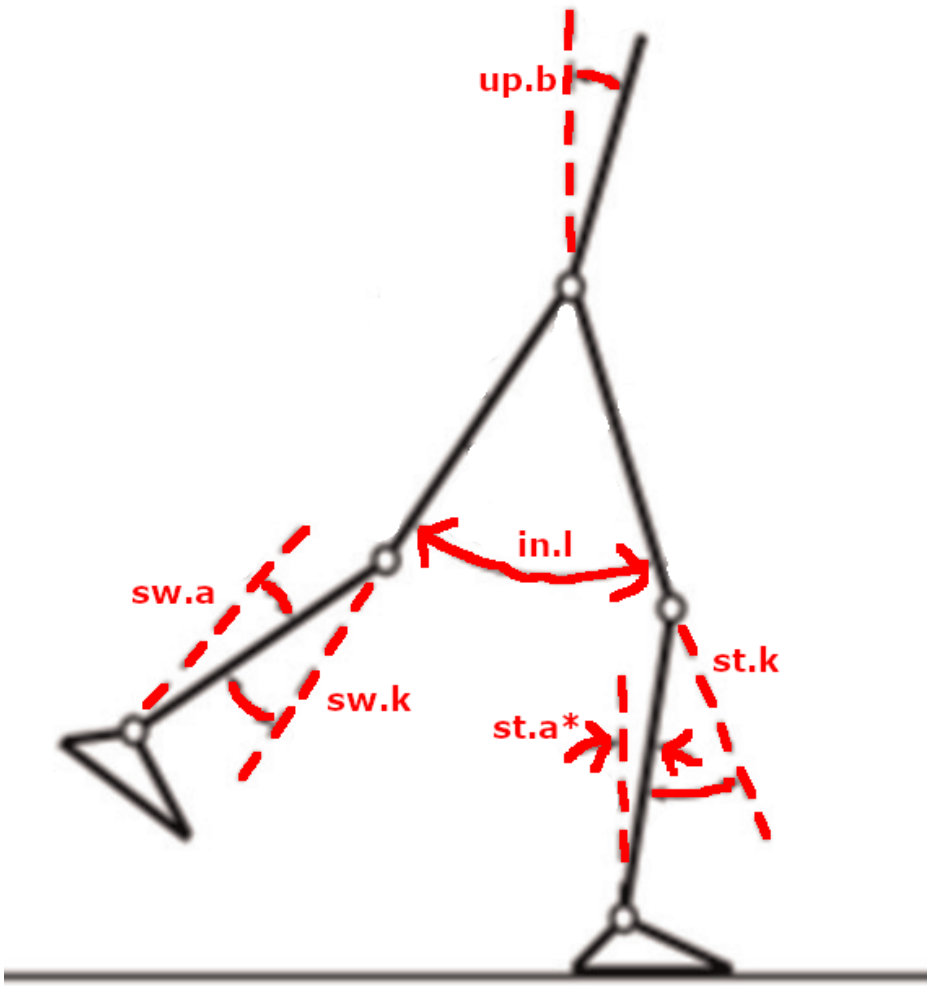

Figure 6.13: **Desired angles in 2D.** Solomon *et al* chose six desired angles for their 2D model. By applying torques to the relevant joints, they adjusted these angles to produce motion. Seen in red, the 2D angles chosen were the upper body, inter-leg, stance knee, stance ankle, swing knee and swing ankle angles. Note that the stance ankle angle was inverted. In 3D, an additional lateral inter-leg desired angle was added.

Default Genotype		
	angle	derivative
upper body	0.03898	0.02870
interleg	0.03702	0.03346
stance knee	0.02128	0.02921
swing knee	0.03908	0.03335
stance ankle	0.02491	0.02218
swing ankle	0.03216	0.03402

Square Base Genotype		
	angle	derivative
upper body	0.08215	0.07830
interleg	0.03332	0.03472
stance knee	0.03051	0.04352
swing knee	0.01694	0.02955
stance ankle	0.02696	0.03952
swing ankle	0.03133	0.02963

Big Tucked Genotype		
	angle	derivative
upper body	0.08375	0.07212
interleg	0.03602	0.03871
stance knee	0.03818	0.01666
swing knee	0.05347	0.05171
stance ankle	0.03068	0.03504
swing ankle	0.02830	0.02282

Figure 6.14: **Genotype Analysis mean input weights tables.** I list the agents' six neural network controllers (for each desired angle), upper body to swing ankle, on the left of the tables. (left to right) The default, *square-base* randomly initialised and *big-tucked* randomly initialised morphologies' mean absolute input weights, each taken over twenty runs of its type. The *square-base*d and *big-tucked* randomly initialised morphologies had higher mean upper body input weights than the default, aligning with them lasting longer. Without averaging, they also had higher upper body input weights than the default according to a statistical significance test. The *big-tucked* randomly initialised morphologies had higher mean swing knee input weights than the default, aligning with the *big-tucked* randomly initialised gait moving its legs faster than the default. These swing knee values were also higher than the default when compared without averaging, according to a statistical significance test.

according to statistical significance tests. The *big-tucked* randomly initialised upper body angle (Mann-Whitney U=1, n1=n2=20, p=1.451e-11, one-tailed, Bonferroni correction for 8 comparisons) and upper body derivative (Mann-Whitney U=8, n1=n2=20, p=4.86e-10, one-tailed, Bonferroni correction for 8 comparisons) were also found to be higher than the default according to statistical significance tests. Figure 6.14 shows the *big-tucked* randomly initialised morphology had higher swing knee input weights than the default, with values found to be greater for swing knee angle (Mann-Whitney U=48, n1=n2=20, p=5.68e-06, one-tailed, Bonferroni correction for 8 comparisons) and swing knee derivative (Mann-Whitney U=48, n1=n2=20, p=5.68e-06, one-tailed, Bonferroni correction for 8 comparisons) according to statistical significance tests. This aligned with the randomly initialised *big-tucked* gait swinging its legs faster than the default.

6.4 Contributions

The contributions made by this work are as follows:

- I evolved bipedal walking agents alongside a square-based morphology based on

the underbelly of a baby albatross, following on from the support polygon balance enforcement rationale of the previous Chapter. The simpler neural network controller in the ODE system was chosen to be more sensitive to morphological changes and require less run-time. This demonstrates this morphology produces agents fitter than the default according to statistical significance tests. This matches my Chapter aim to use the square-based morphology to increase fitness and my general aim to utilise morphology to simplify the bipedal task.

- I also evolved bipedal walking agents alongside a morphology based on the underbelly of a baby albatross featuring tucked knees, following rationale of more motion from the previous Chapter. The simpler neural network controller in the ODE system was chosen to be more sensitive to morphological changes and require less run-time. This demonstrates this morphology produces agents fitter than the default according to statistical significance tests. This matches my Chapter aim to use the tucked-knees morphology to increase fitness and my general aim to utilise morphology to simplify the bipedal task.

- I evolved these morphologies alongside the neural network controller, creating a second morphological genotype for leg length and seeding both with previously high-fitness agents. Randomly initialising these morphology and neural network controller genotypes instead achieved fitness up to four times higher. Whilst this does not achieve my Chapter aim of returning to standard proportions with increased fitness, this is still in line with my general aim to utilise morphology to simplify the bipedal task.

- I also evolved a *big-tucked* randomly-initialised morphology agent with higher input weights for its upper body and swing knee neural network controllers according to a statistical significance test. This agent featured an extremely fast gait as a result. This matches my aim to increase walker fitness.

6.5 Discussion

An interpretation of the results seen in previous Chapters and again here is the novelty over efficiency rationale. The albatross morphologies are accompanied by a complete removal of the torque constraint throughout walker lifetime. In addition, the *big-tucked* morphology is based on the control cost reduction enhancement of the previous Chapter. Whilst not a literal reduction of efficiency enforcement, the *big-tucked* morphology instead enables more motion with its modified bird legs, aiming

to lift the walker up as though it were a scissor lift. The albatross morphologies produced higher fitness according to statistical significance tests. This could be interpreted as further proof of the success of this philosophy. This could again lead to more complex behaviour in future evolutionary works that adapt the philosophy. This also could demonstrate the robustness of the idea further, as it has been translated into morphological adaptations as well as direct enhancements.

Another interpretation of the results seen here is the connection between morphology and control in the bipedal walking task. The walkers initialised with a random morphology genotype and a random network genotype featured higher fitness than the walkers initialised with both preloaded ones according to statistical significance tests. The highest-fitness random agents had evolved much longer legs to take bigger steps, whilst the highest-fitness walkers with a preloaded neural network controllers morphologies barely changed from their starting point. This can be interpreted as the walking task having a strong link between morphology and control - the neural network controller local optima was surrounded by such a drop in fitness in morphology space that changing it even slightly resulted in no longer being selected by the genetic algorithm. This leads me to consider using more varied mutation rates across neural network controller and body in future works, in order to overcome this gap. This could also provide further evidence that bipedal agents are more strongly linked to their morphology than quadrupedal ones, due to their need for increased balance.

Despite promising results, something to also note could be the choice of physics engine. Methodologies based around a simulation in the Mujoco engine from Chapter 5 may not be applicable to the ODE engine. Several studies described in Chapter 2 have investigated capabilities of physics engines for different tasks [7] [29], both of which demonstrated that different engines are designed for different aspects of simulation, such as numerical accuracy or constraint enforcement. The albatross morphologies require collision between a supporting sphere component and the ground. Erez *et al* demonstrated that ODE performs poorly at collision management by dropping a set of capsules at the same time. On the other hand, the albatross morphologies also require the larger sphere components to be constrained tightly to the body with minimal error, something Boeing *et al* demonstrated ODE to be accurate for with a long chain of linked bodies, albeit with a certain integrator setting not used here. Despite this, in the end, ODE was chosen as it matched the work seen in Chapter 4, a system

with a simple neural network controller, ideally meaning results could be clearly attributed to a change in morphology. Comparatively, whilst Mujoco performs better in a large number of trials, the walker used in Chapter 5 is much more complex in neural network controller, and as such results could be partially attributed to neural network controller adaptation instead. In addition, whilst the changes are similar in nature they vary in implementation. Chapter 5 uses fitness function changes and fail conditions, whilst here I use a change in morphology. I hope these motivations outweigh any differences potentially incurred through the change from Mujoco to ODE.

6.5.1 Acknowledgements

My code is based on the work of Solomon *et al.* It is available via GitHub at *https://github.com/benjack795/solomon-reimp* for exact specification and future reproductions.

Chapter 7

Evolving Bipedal Walkers for Turning and Pursuit

7.1 Chapter Aims

- In this Chapter I aim to produce pursuit-turning behaviour in the evolved albatross walkers from the previous Chapter. This will follow on from the previous turning behaviour investigated in Chapter 4 and to my knowledge, be a unique behaviour in evolved bipedal systems, approaching the level of behaviour seen from Heess *et al* [35]. This is in line with my aim of producing complex behaviour.

7.2 Methodology

7.2.1 Rationale

After producing higher fitness via the *big-tucked* albatross morphology in the previous Chapter according to statistical significance tests, I aimed to use this robust morphology to extend the single-turn behaviour produced in Chapter 4 into multi-turn behaviour, in pursuit of a moving target point. As mentioned in Chapter 4, I believe turning is the next logical task to aim for after forward walking, as it is required to move freely in 3D as a bipedal agent. Multiple turning extends this further and produces more complexity. Choosing multiple turning as the next behaviour to produce is also useful for testing systemic response to sensory input. This can be seen from Reil *et al*, who produced lateral turning behaviour following on from forward walking in their CPG-based system in order to test sensory response [64].

Unlike the previous Chapter, this new task mainly requires a change in fitness function

and not morphological changes. Thus, it may seem prudent to return to the Salimans *et al* deep learning for the relevance it provides. However, in the end I decided not to do this for a number of reasons. Firstly, turning behaviour had already been evolved on the Solomon *et al* system in Chapter 4. This meant I already had an estimate of the capabilities of the system to evolve turning based behaviours, compared to the other system, and was more likely to produce complex results than an untested system. In addition, I want morphology to remain a factor in future walkers alongside complex evolved behaviour, something the Salimans *et al* system is less suited for with its complex neural network controller. Not many works utilise morphology alongside evolved bipedal locomotion, and if I can achieve this, it can help to further demonstrate the value of morphology for producing complex behaviour. An alternative would be to attempt the new fitness function on the Salimans *et al* system, but as mentioned, this would be less useful and less likely to produce results immediately.

7.2.2 Pursuit-Turning

In order to produce pursuit-turning behaviour with the *big-tucked* walkers, I adjusted the fitness function of the system, carrying on from the rationale in Chapters 4 and 5 of fitness being the most direct way to influence an evolutionary system. In addition, I re-enabled the 3D inputs, joints, and lateral hip neural network controller. The 3D inputs were re-enabled to provide additional control in the lateral dimension as in the original turning experiment in Chapter 4. Previously, fitness was the distance travelled forwards in the x dimension. To reward pursuit, I modified the function to be a cumulative value for closeness to a target point per time-step. Precisely, this was modelled as the total of 15 simulation units minus the distance from the target point, divided by 10000, for every time-step. The final fitness function can be seen in Equation 7.1, modelling f as fitness, t as the current time-step, d as the distance from the walker to the target point at a given time-step, and m as the number of moves the walker has achieved. The simulation time-step size was set to 0.005s; walkers were allocated a 60s lifetime; this meant there were 12000 time-steps in a walker's lifetime. The walkers were also equipped with an additional network input. This was a sensor, which gave a reading from 1 to 8 based on the target point's heading from the walker's position. In order to achieve this, the difference between the walkers heading and the heading of the goal point from the walker is calculated and then the sensor value is assigned based on which of the 8 segments of a complete rotation the resulting angle falls within. I initialised the target point 4 simulation units in front of the walker, requiring a brief walk forward before turning could begin. When

the walker's body (not counting the large sphere attachment) got too close to the target point (closer than 1 metre), the target point would move a quarter of the given wavelength forward in x and a given amplitude either to the left or the right, seen in Figure 7.1. 1 metre was chosen for the distance margin as simplest unit for a baseline experiment. However, this resulted in a cheating behaviour - getting as close to the target point as possible and stopping, without getting so close that it moved it. To remedy this, I added a large bonus to the cumulative score for each time the agent moved the target point, such that an agent that had moved the target point would always score higher than one that had not. Precisely, this was an additional 15 multiplied by the number of moves the target point had made, again divided by 10000 for scale, every time-step. This bonus would thus be zero for walkers that had not moved the target point, and would then increase in a linear fashion each time the target point was moved.

$$f = \sum_{t=1}^{12000} \frac{(15 - d(t) + 15m)}{10000} \tag{7.1}$$

The parameters in Equation 7.1 are chosen for a variety of reasons. The first terms in the equation are 15-d(t). These terms were used to produce a distance-based value for the agents, scoring them on how close they were to the target point. 15 was chosen as as a value much larger than the initial distance to the point, almost 4 times larger. The chances of the walker being more than 15 simulation units away were slim given the distance walkers had travelled forward in Chapter 4 around the same distance. Any larger and the values produced would be large and disproportionate compared to fitness values in Chapter 4, any smaller and they could be negative, producing potentially confusing figures. In this way, distance at time-step d(t) could then be subtracted from 15 to produce a value that increased based on how close the walker was to the point, and would be very unlikely to be negative. A possible alternative to this methodology would have been to sort walkers inversely to their fitness, but this would have made the function more complex to scale when the bonus term was added later on. The next term in the equation was 15m. This term was chosen as a bonus value for each time the point was moved, ensuring a score with more moves would always produce a higher value than one with less moves, with a bonus of 15 for each move being equal to the highest score possible from the distance term. Alternatively, the number of moves could have been used as a fitness score, but this would lack differentiation in strategies with the same number of moves, which could vary vastly in behaviour. This in turn could cause the system to struggle to evolve more nuanced

behaviours. The final term in the function is 10000. This was chosen in order to scale the values to a more appealing range visually. As the function sums fitness values across the 12000 time-steps in a walker's lifetime, values could easily reach this order of magnitude and higher. 10000 was chosen as the scaling value as it was the nearest order of 10 to 12000. Alternatively, 12000 could have been used directly to produce something closer to an average fitness score, but this was decided against to produce values that would require less rounding.

In addition there are two other parameters that are explained here. The first of these were the coordinates chosen for the target point from the initial walker position. The position chosen was [4,0], a point 4 simulation units forward from the walker's initial position. This point was chosen in order to allow the walkers to settle into a forward gait before turning, which is easier then turning immediately. The value of 4 specifically was chosen as walker legspan was around 1.2. As such, a distance of 4 would equate to around 4 steps, or 2 full gait cycles. Alternatively a distance of two could have been used for 1 complete cycle, but due to the difficulty of the task, 2 was chosen, in case the first was achieved by cheating in some way.

The final remaining parameter is the range chosen for the sensor input added to the walkers. Walkers receive an input from 1 to 8 based on where the point is in relation to the walkers heading. 1 to 8 was chosen as most compasses have 8 main directions on them, so it made sense to split the sensor range into 8 regions. A higher sensor resolution would have made the input more complicated than necessary, making it more difficult for the neural network controller to map it to behaviour. Alternatively, a lower input, for example 1 to 4, would have lacked the information to discern between a slight turn away, something achievable by leaning whilst stepping, and a large turn away, requiring a full turn from several sideways motions and steps. These two tasks require vastly different amounts of motion, and as such it was felt they needed differentiation.

As mentioned above, when the walker agent is less than one simulation unit from the target point, it moves away in a zigzag pattern, crossing back and forth across the x-axis whilst increasing in x. This pattern can be seen in Figure 7.1 below. In order to follow the point, the walker must exhibit turning behaviour. There are two main parameters in this pattern: wavelength and amplitude. Wavelength is the distance travelled forwards in X after one complete iteration of the zigzag pattern. It

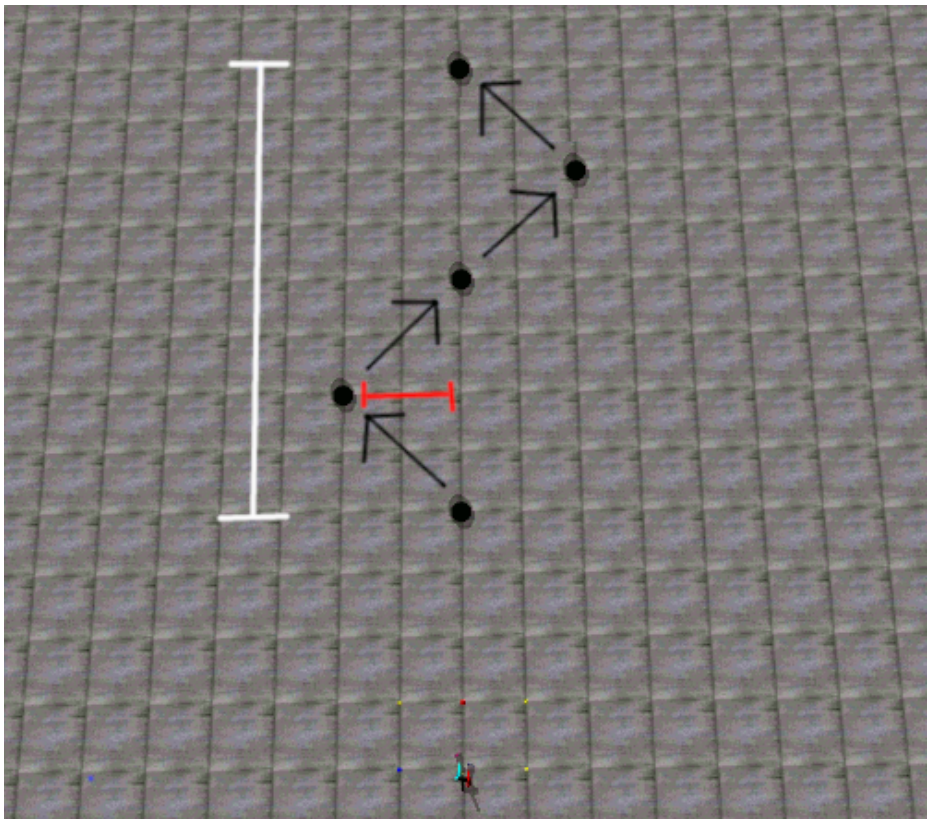

Figure 7.1: **The turning task layout.** The walker (bottom-middle) had to follow the black target point, starting from a little in front of it, as it moved back and forth across the x-axis and away, forcing it to turn back and forth. I varied the amplitude (red) and wavelength (white) of the pursuit pattern. Fitness was measured as cumulative inverse distance to the target point per time-step, with an added bonus for the total amount of times the target point had been already moved.

is named wavelength as it represents the full length of the pattern, viewing it from above as a wave. This full pattern is a move to the left, then back to the center, then right, and then back to the center again. Wavelength is represented by the white line in Figure 7.1. The other parameter is the amplitude, which is the maximum distance the zigzag pattern travels laterally in y away from the x-axis, in both directions. It is referred to as amplitude as it is the peak value the pattern travels in y, when viewing the zigzag pattern as a wave from above. Amplitude is represented by the red line in Figure 7.1. To analyse turning behaviour further, I produced multiple perturbations of this system with varying amplitude and wavelength parameters.

7.3 Results

I tested a range of amplitude and wavelength combinations to produce pursuit-based turning behaviour. For each permutation I launched 20 runs of the system, each with a population of 150 for 500 generations, chosen as the largest number of generations for a large enough population that could be executed within a reasonable runtime. Twenty was chosen as the generally accepted minimum sub-group sample size for quantitative research. I assessed how many times each walker made the target point move away from it, which is henceforth referred to as the number of "moves". Walkers were terminated after 60s.

Figures 7.2 and 7.3 show the maximum number of moves from each set of 20 for each permutation of the amplitude and wavelength parameters. When varying amplitude, there was little variance at most amplitudes besides amplitude 1, in which the smallest wavelength, 4 featured a high number of moves. This can be explained as being due to the proximity required to move the target point. The target point moved away when walkers became less than 1 metre from it. Therefore a walker with a amplitude of 1 could walk straight down the middle with little deviation and move the target point regardless. All wavelengths were still able to move the target point at least three times (equivalent to two turns) at high amplitudes. No permutation dropped below 2 moves (equivalent to one turn). When varying wavelength, the shortest amplitude was able to move the target point more than five times at multiple wavelengths, decreasing in moves as wavelength increased. The runs with a amplitude value of 2 were able to move the target point five times (four turns) at wavelength 16. All permutations achieved at least two moves.

Figures 7.4 and 7.5 show the mean number of moves from each set of 20 for each permutation of the amplitude and wavelength parameters. When varying amplitude, only the lowest amplitude produced a mean of more than four moves. The lower wavelengths were again the slowest to fall into the lower values as amplitude increased. Almost all the values were below two: a much lower set overall when compared to the maximum values. When varying wavelength, again the lowest amplitude produced the highest number of moves, and the lower amplitudes in general featured higher moves than the rest at lower wavelengths. This large gap between maximum values and mean values demonstrated the difficulty of the task through the scarcity of high moves.

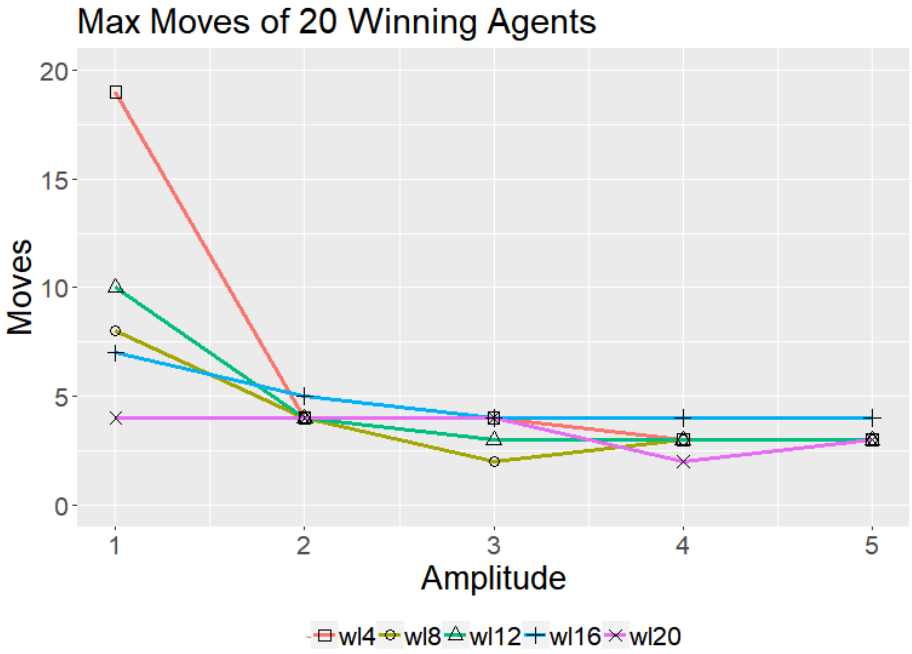

Figure 7.2: **Maximum Moves per Amplitude.** The maximum number of moves for each set of 20 agents for each wavelength parameter, at amplitudes 1,2,3,4 and 5. There are no obvious trends besides some amplitude 1 runs having higher maximum values than the rest, suspected to be due to the nature of the task - a distance less than 1 is required to move the target point, meaning very little turning is required at amplitude 1.

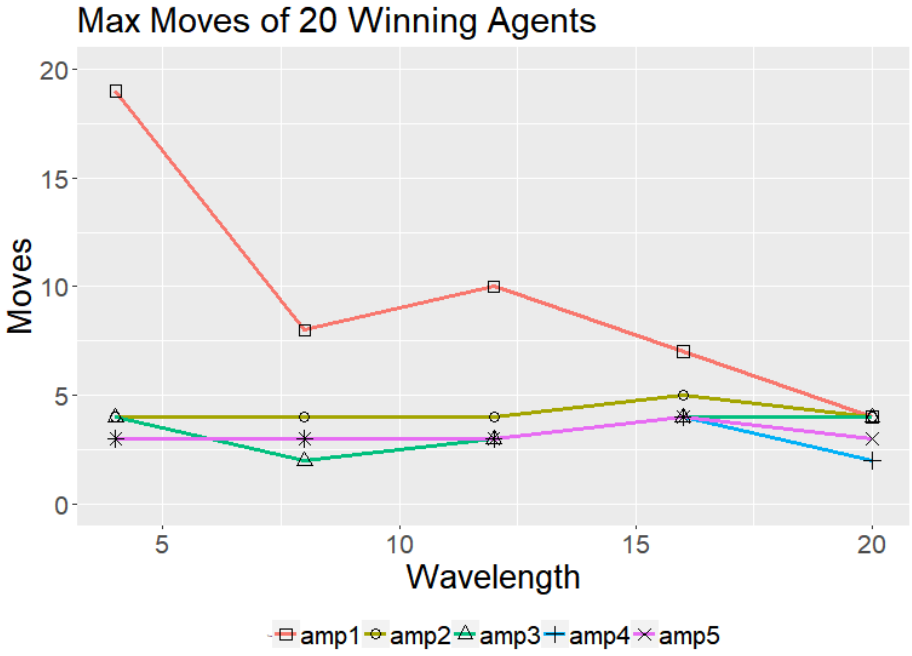

Figure 7.3: **Maximum Moves per Wavelength.** The maximum number of moves for each set of 20 agents for each amplitude parameter, at wavelengths 4,8,12,16 and 20. The amplitude 1 runs slowly decrease in maximum moves as wavelength increases; the others remain the same throughout.

I also analysed the walking paths taken by agents from the the runs with a wavelength of 16. I chose this wavelength for further analysis as it featured multiple double-turn behaviours. The path taken by the best of each of these sets of 20 runs is seen in Figure 7.6. The amplitude 1 and 2 agents were able to make more than two turns; 1 in particular had an impressive wave-like arc, despite requiring little turning motion overall due to the low amplitude. The best evolved agent at amplitude 1 also appeared to be heading in the correct direction to continue its gait, as though limited only by the time constraint. The remaining higher amplitudes were also all able to make 2 turns, as seen by their initial turn toward and then second turn away from their peak values. Interestingly, amplitude 4 adopted a different turning strategy to the other four, bouncing on the spot to turn without moving, as seen from the sharp point in its curve.

Finally, I display several gaits from the higher fitness wavelength/amplitude perturbations, featuring several multiple-turn behaviours. These were a1w4, a2w4, a3w4, a1w16, a2w16 and a3w16. The lower amplitudes of the 4 wavelength were chosen for

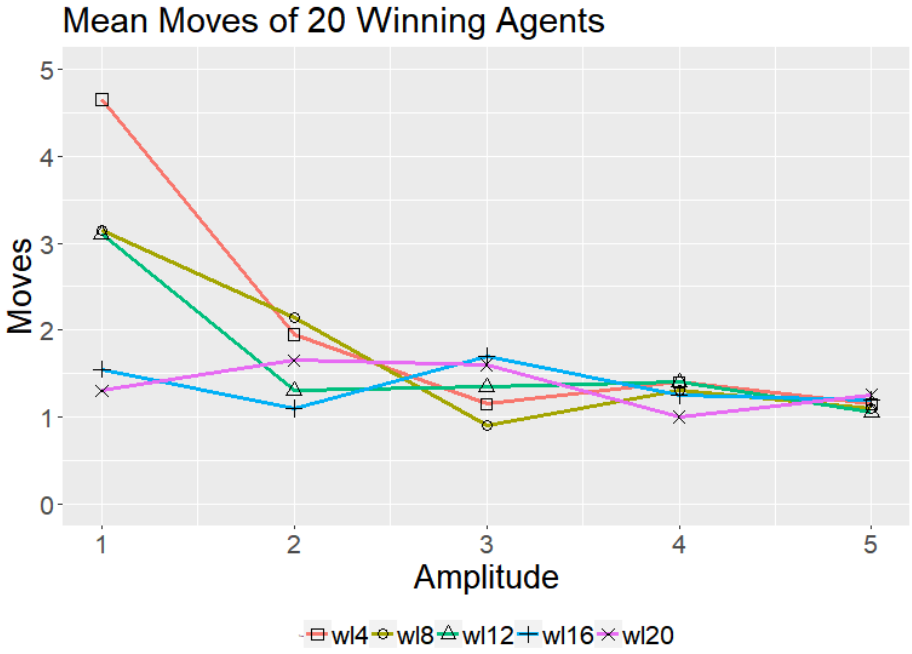

Figure 7.4: **Mean Moves per Amplitude.** The mean number of moves for each set of 20 agents for each wavelength parameter, at amplitudes 1,2,3,4 and 5. The lower wavelengths produce higher moves at lower amplitudes, then decrease in moves until they match the others at higher amplitudes.

analysis as they exhibited high maximum and mean numbers of moves. Meanwhile the 16 wavelength lower amplitudes were chosen for analysis as they exhibited multiple interesting double-turning behaviours. Figure 7.7 shows the gait of the a1w4 perturbation. This agent walked forward for a long distance, forcing the target point to move several times without having to exhibit large turning motions, due to the low amplitude. Eventually it veered off to one side. Figure 7.8 shows the gait of the a2w4 perturbation. This agent stumbled for the first part of its lifetime, before settling into a stepping motion. It then turned away to the left and back again toward the centre and then the right. It was unable to turn left again and veered off. Figure 7.9 shows the gait of the a3w4 perturbation. This agent made a wide turn to the left and then back to the right through the centre, before being unable to turn back again. Figure 7.10 shows the gait of the a1w16 perturbation. This agent turned slightly more than the agent with the same amplitude and a wavelength of 4 did at the lower amplitude, eventually stopping after walking forwards for a long period of time. Figure 7.11 shows the gait of the a2w16 perturbation. This agent turned away from the x-axis to the left and then back again, across the centre, before remaining on the right side until the end of its lifetime. It stumbled less than the a2w4 perturbation. Figure

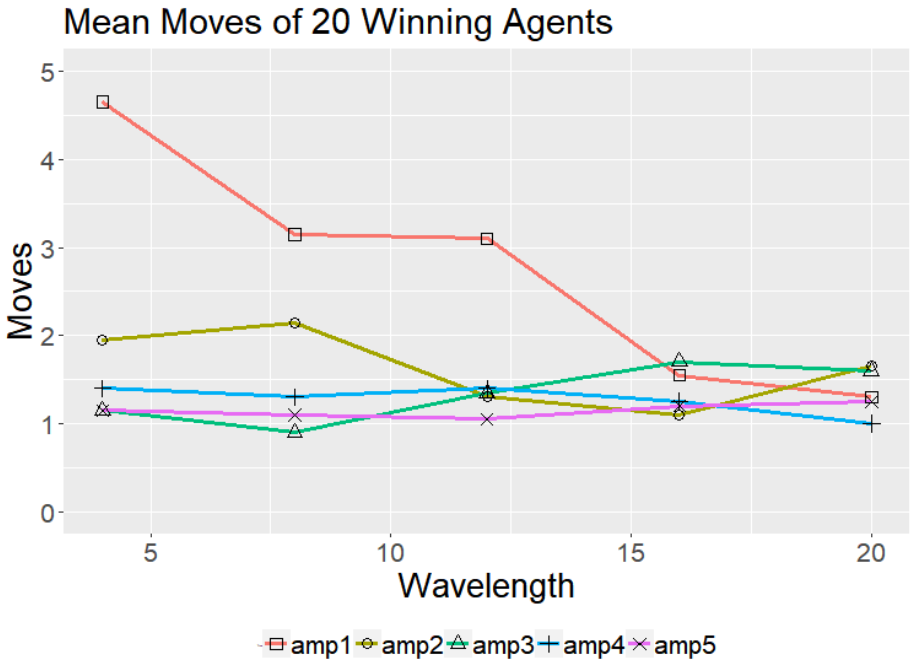

Figure 7.5: **Mean Moves per Wavelength.** The mean number of moves for each set of 20 agents for each amplitude parameter, at wavelengths 4,8,12,16 and 20. The amplitude 1 runs mean value starts notably higher than the others and decreases to match them as wavelength increases.

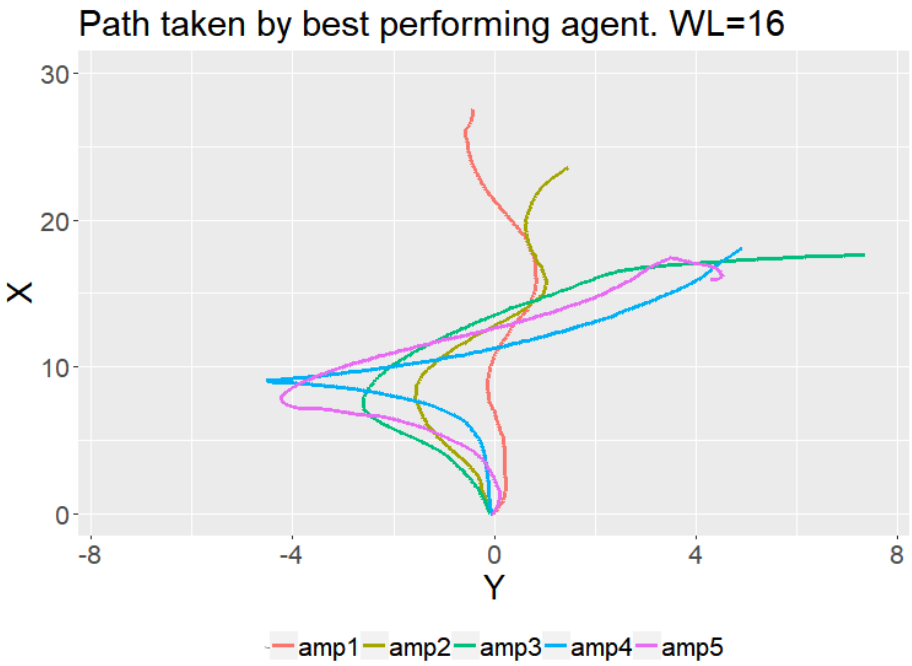

Figure 7.6: **The highest fitness gaits at the 16 wavelength.** The highest fitness gait from each amplitude's set of 20 runs, with a wavelength parameter of 16. Each line represents the walker's path viewed from above, starting from 0,0 and following the target point in a zigzag motion. The amplitude 1 and 2 agents were able to make the most turns, with the lowest amplitude managing five. The remaining higher amplitudes were also all able to make two turns, as seen by their initial turn toward and then second turn away from their peak values. Amplitude 4 bounced on the spot to turn without moving, as seen from the sharp point in its curve.

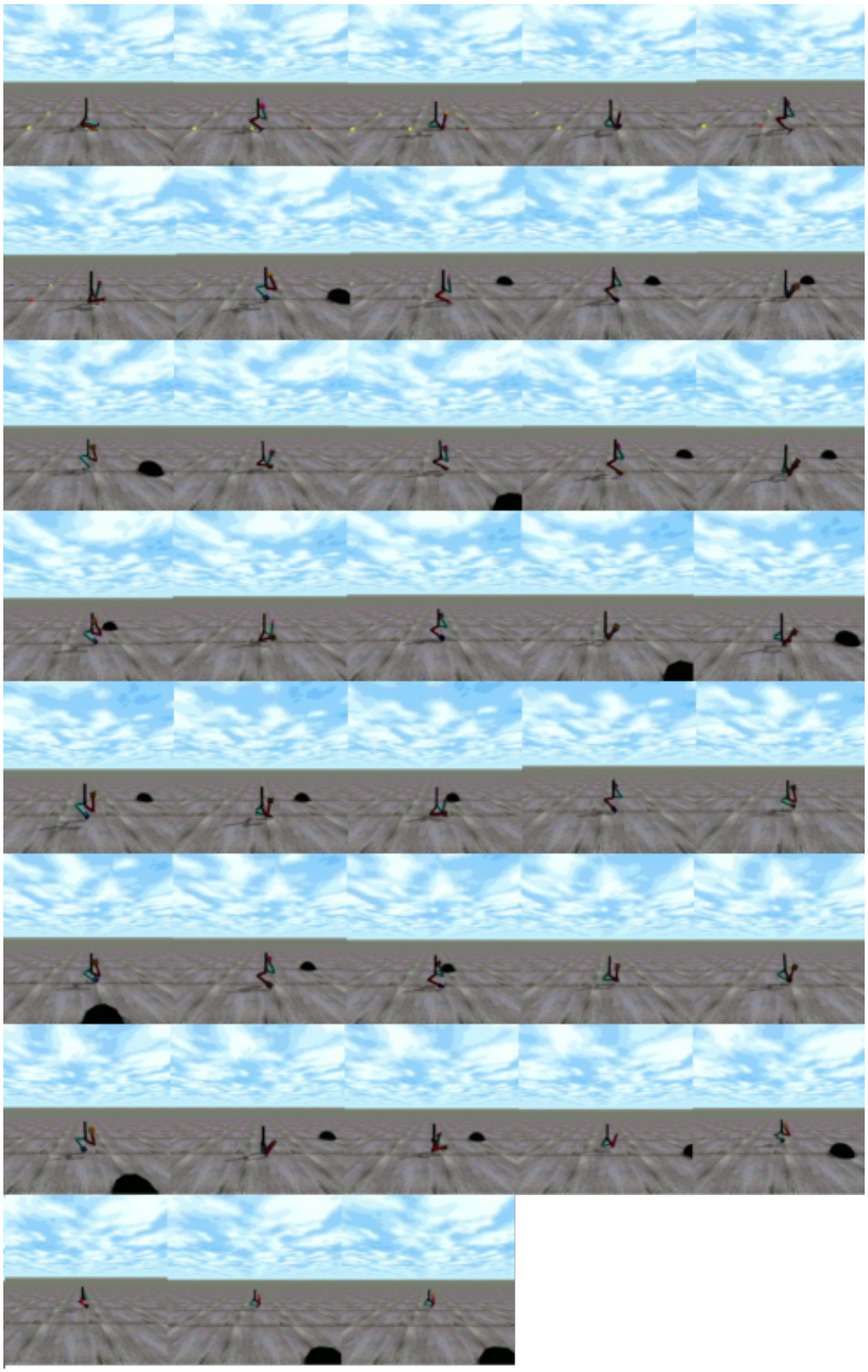

Figure 7.7: **(left to right, top to bottom) The highest fitness gait for the a1w4 perturbation.**

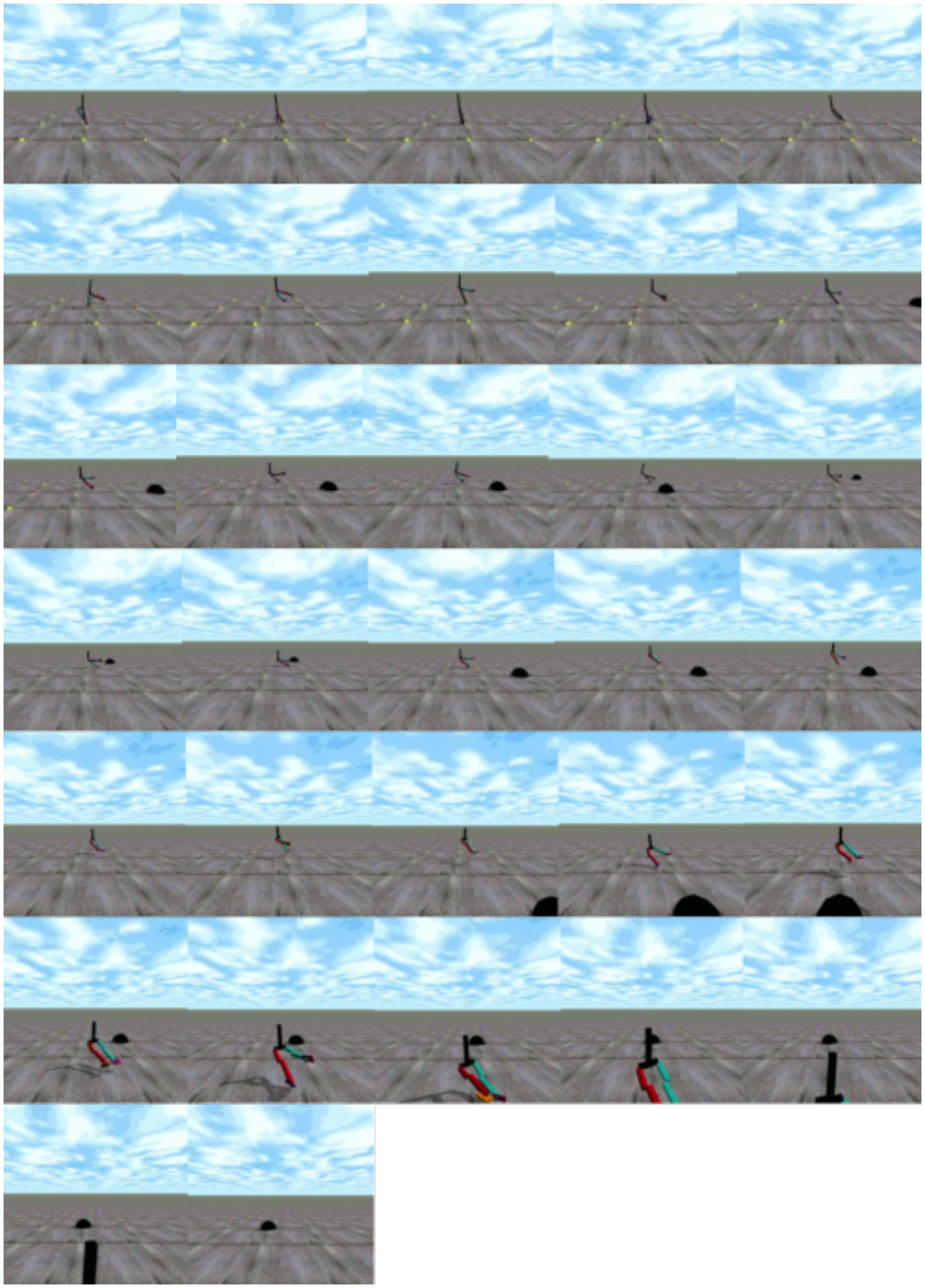

Figure 7.8: **(left to right, top to bottom) The highest fitness gait for the a2w4 perturbation.**

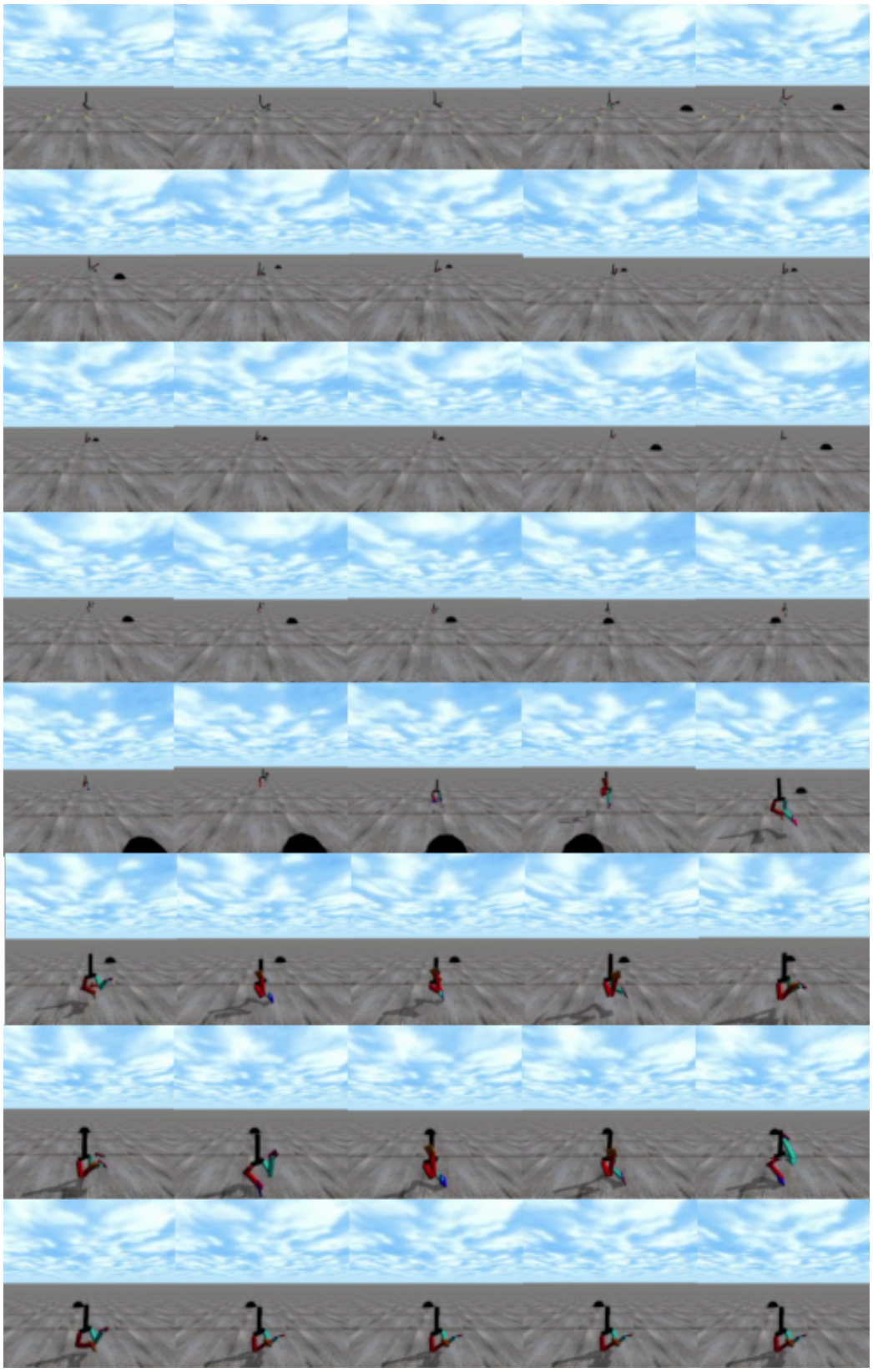

Figure 7.9: **(left to right, top to bottom) The highest fitness gait for the a3w4 perturbation.**

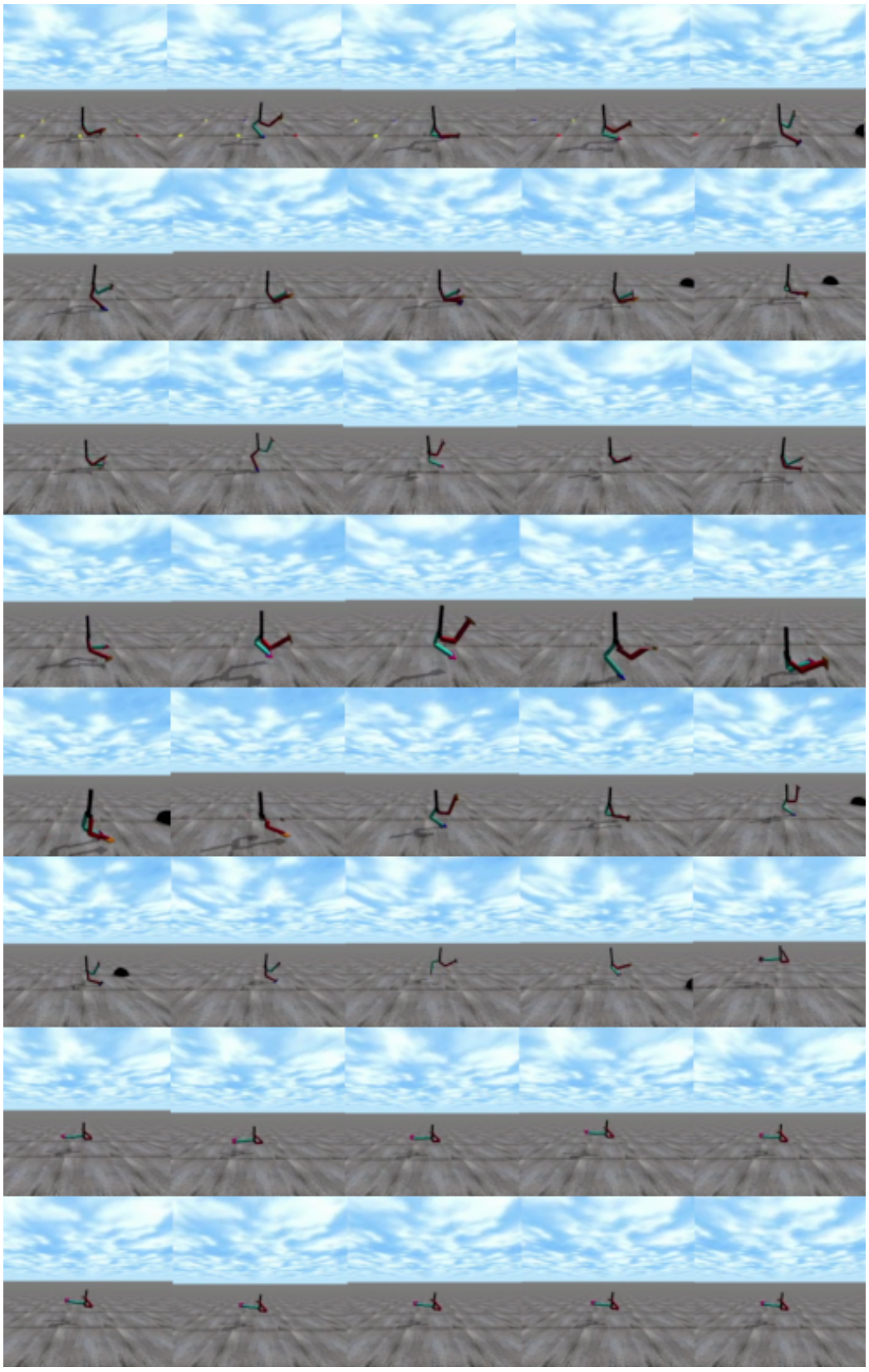

Figure 7.10: **(left to right, top to bottom) The highest fitness gait for the a1w16 perturbation.**

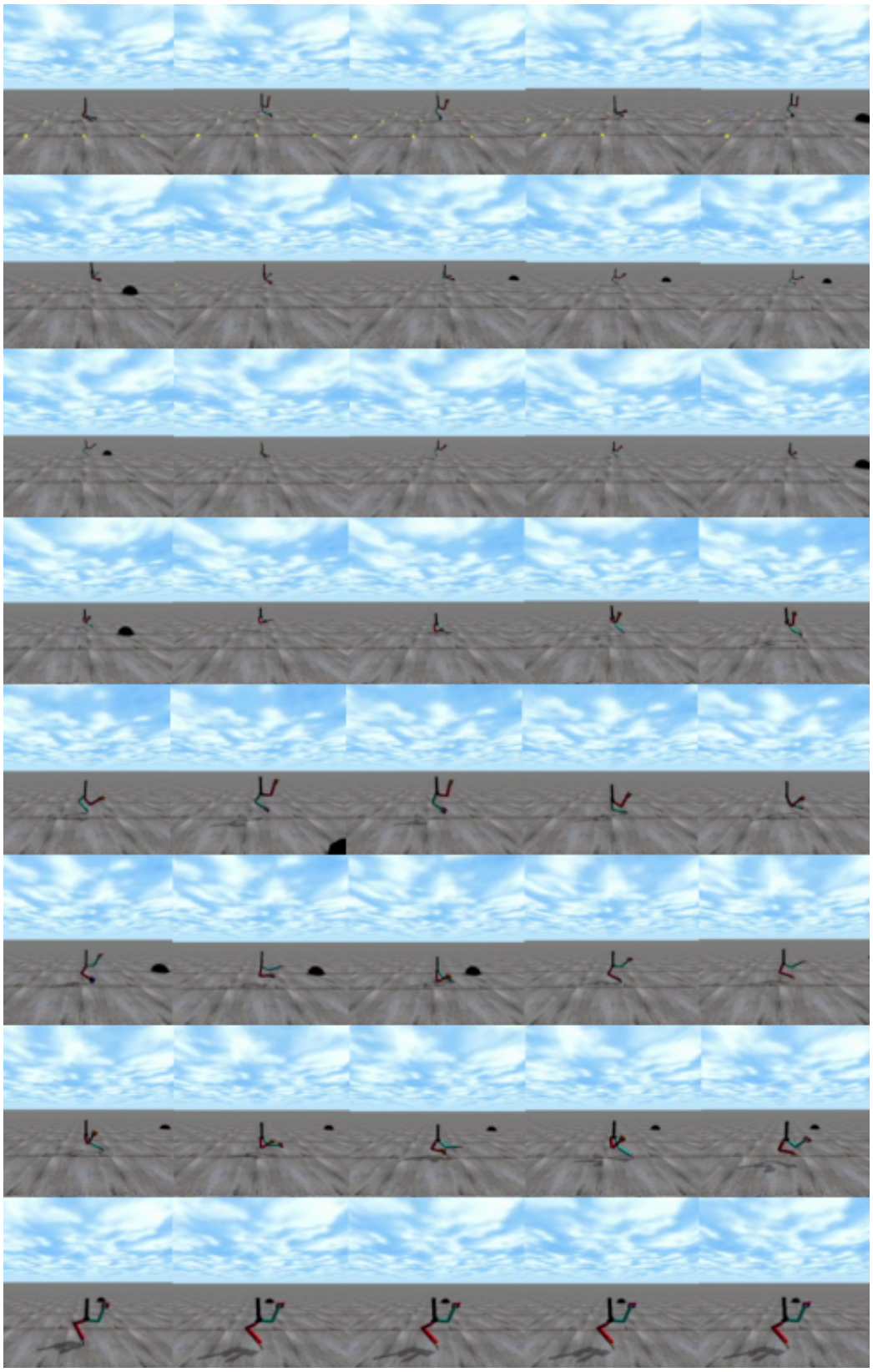

Figure 7.11: **(left to right, top to bottom) The highest fitness gait for the a2w16 perturbation.**

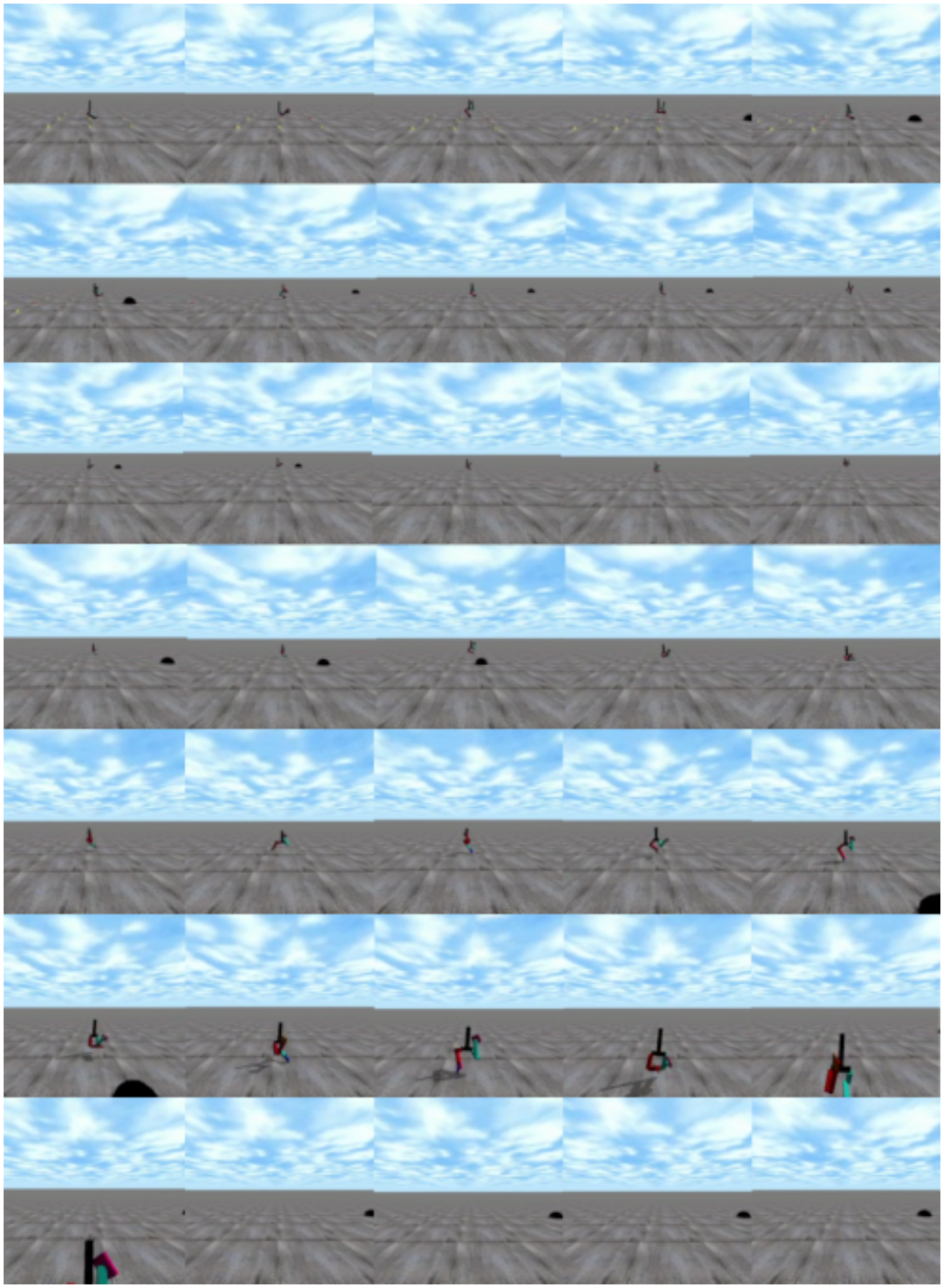

Figure 7.12: **(left to right, top to bottom) The highest fitness gait for the a3w16 perturbation.**

7.12 shows the gait of the a3w16 perturbation. This agent made a wide turn away from the axis and back again before veering off. This perturbation matched the a3w4 perturbation.

7.4 Contributions

The contributions made by this work are as follows:

- I evolved walkers in my enhanced re-implementation of the Solomon *et al* Linear Reactive system from Chapter 4, together with my high-fitness *big-tucked* albatross morphology from Chapter 6, on a pursuit-based turning task, designed as an extension of the single-turn behaviours produced in Chapter 4. This was chosen instead of the Salimans *et al* system to further demonstrate the utility of morphological changes by retaining them alongside complex behaviour. Explicitly, this demonstrates the pursuit-turning task produces walkers capable of producing four consecutive turns, and two turns at high amplitudes, in pursuit of a moving point. The pursuit-based turning behaviour evolved here is also an example of complex behaviour that, to the best of my knowledge, has not been attained previously in evolved bipedal walkers. This was difficult previously as traditional bipedal agents have a high center of mass and feet close together, making them tricky to balance alongside other complex tasks. The behaviour resembles the the slalom behaviour produced by Heess *et al*. This matches my Chapter aim to produce pursuit-turning behaviour and general aim to produce complex behaviour.

7.5 Discussion

A possible interpretation of this work alongside previous Chapters is the importance of using fitness functions to enhance evolutionary systems. In order to evolve turning behaviour, I applied a new fitness function based around cumulative distance, similarly to Chapter 4. This resulted in walkers exhibiting as many as four turns in pursuit of a point. This could be interpreted as further proof of the value of this philosophy. This could in turn lead in to further evolutionary works using this philosophy to produce more complex behaviour. This could also suggest fitness function modifications are the best method to produce turning behaviour, as this is the second Chapter to succeed in producing turning behaviour with a new fitness function.

Another possible interpretation of this work is the implications of producing behaviour similar to that of Heess *et al.* The zigzag-like turning behaviour produced by my turning task is very similar to the their slalom behaviour [35]. This could be interpreted as a step towards matching their bipedal behaviours produced through reinforcement learning. This could in turn lead to further works designing tasks similar to the one featured here to create more complex bipedal behaviour seen from their work, such as running or climbing. This could strengthen the rationale mentioned in Chapter 4 and here further, that turning is the next task to aim for after forward walking, as the successful walker agent system was able to produce turning behaviours easily with a new function, connecting the two tasks.

A second more important conclusion I can draw from this interpretation of results is the renewed use of evolutionary systems. Evolutionary systems have historically struggled to match the behaviours of reinforcement learning on bipedal tasks, if not in fitness, then definitely in gait and behaviour. The behaviour produced here could begin to lay out a road-map for future evolutionary systems to build upon with similar techniques, to achieve more behaviours produced by reinforcement learning (and perhaps some not yet produced). Given the nature of evolutionary systems to be easily incremented through several tasks, this could be a matter of seeding the population with high-fitness turning agents and utilising a new fitness function. This would also benefit from the additional biological insights of evolutionary processes unavailable to reinforcement learning, potentially teaching us more about the evolution of bipedalism in the biosphere.

A final interpretation of this work was the similarity in results amongst walkers at higher amplitudes. Whilst there were no trends across the entire dataset described above, Figure 7.2 shows the max number of moves for each permutation by amplitude experiences little variation in the higher amplitudes compared to the lowest one, with ranges of around 2.5, compared to the highest range at around 15. This pattern is also seen much more noticeably in Figure 7.4, the mean number of moves for each permutation by amplitude. Here the range of values gets visibly smaller as amplitude increases, in a linear fashion, from a range of around 4 to a range of around 0.5. Finally in Figure 7.6, amplitudes 3, 4, and 5 all exhibit similar trajectories compared to 1 and 2. The top three amplitudes all make a single sharp turn and then continue their heading, not turning a second time.

This could be interpreted in several ways. Firstly, as described in the previous section, this could be due to the lesser amplitude walkers not having to turn as much as the distance to move the point matches the lowest amplitude, 1. If this was the case, it would demonstrate the importance of the threshold distance parameter to the turning task, linking it directly to the style of turning behaviours that emerge. The lack of variety at high amplitudes could also be due to the nature of longer turns requiring more walking and balance. This could demonstrate that turning as a behaviour will always produce less variety at higher amplitudes, and that turning tasks may benefit from the assistance of a novelty methodology to produce larger turns. The interpretation that I find the most interesting however, is a potential ideal gait difference between the upper and lower amplitudes. When the lower amplitudes need to turn to face the point, the difference in heading is much smaller than at a higher amplitude. As such, the walkers could achieve a turn by simply leaning one way whilst walking, compared to having to make several lateral stepping motions for a sharper turn. This would explain the divide between the upper and lower amplitudes and demonstrate that different turning behaviours are optimal for different styles of turn for this task. Whilst the gait figures above do not necessarily reflect this behavior, the trends indicated from the results figures cover the entire sets of runs for each perturbation, and could still validate this interpretation. This could lead to new ways of approaching complex bipedal behaviour. The gap between these two distinct groups could be bridged with incremental evolution, starting with high fitness agents for the lower amplitude turns and seeing whether or not incrementing up to the higher amplitude turns is easier than evolving the higher amplitudes outright. If this were a success, it could then lead then lead to a break down of other bipedal tasks into easier incremental sub-tasks and potentially a new wave of complex behaviour in evolved bipedal agents.

7.5.1 Acknowledgements

My code is based on the work of Solomon *et al.* It is available via GitHub at *https://github.com/benjack795/solomon-reimp* for exact specification and future reproductions.

Chapter 8

Incrementally Extending the Turning-Pursuit Task

8.1 Chapter Aims

- In this Chapter I aim to improve the pursuit-based turning behaviour task from the previous Chapter using incremental evolution, after the pursuit-turning task produced two separate classes of behaviours that could potentially be incremented between. This could mean higher amplitude behaviours can be produced easier incrementally than from scratch. This is in line with my aim to highlight the merits of evolutionary processes and also my aim to produce complex behaviour.

8.2 Methodology

8.2.1 Rationale

In the Solomon *et al* system, the highest fitness 2D neural network controllers were incrementally loaded as the starting population for the 3D engine neural network controllers. This proved to be a success, and even though I enhanced the system without prior bootstrapping in Chapter 4, the strength of the methodology was still notable. I also observed success from works such as Stanton *et al* [79] and Mouret *et al* [58] for incremental evolution on complex tasks. Finally, I decided to produce a final in-depth investigation of incremental evolution after the results of my most recent successful task, novel pursuit-turning behaviour. This task evolved two distinct behaviours for low and high amplitude, leading me to believe I could use incremental evolution to bridge the gap between them, and therefore potentially produce high amplitude behaviours easier than evolving them outright by scaling up. This also

would enable me to fulfil my general aim of highlighting the merits of evolutionary techniques over other methods for bipedal behaviour.

8.2.2 Incremental Pursuit

Achieving this bridge involved varying the amplitude parameter of the pursuit-turning task from the previous Chapter over each generation of the genetic algorithm. Following Equation 8.1, modelling g as the current generation, G as the total number of generations and M as the maximum amplitude, I linearly scaled amplitude a per generation up to the maximum value, over a period of half the total generations. I then maintained maximum amplitude for the remaining generations. In the previous Chapter, one of the main theories considered from the results of different perturbations of amplitude and wavelength was the contrast between high and low amplitude behaviours. Walkers at lower amplitudes were exhibiting different trajectory styles to those at higher amplitudes, suggesting the ideal method of locomotion is different between the two. The equation in 8.1 scales the amplitude value linearly up to a maximum at half generations, to see if a slow transition between the two is easier than evolving the more difficult higher amplitude behaviours outright. Half generations was chosen as the switching point in order to provide an equal amount of both incremental linear scaling and standard evolution, allowing for the system to maintain the best parts of each with an equal balance in the run-time available. I speculate that by evolving walkers at smaller amplitudes first, it may be easier to evolve turning behaviour at higher amplitudes later. I applied this to all the same perturbations as the standard version of the task to accurately assess any improvements in fitness or behaviour.

$$a(g) = M \cdot \min\left(1, \frac{2g}{G}\right) \tag{8.1}$$

8.3 Results

I tested a range of incremental amplitude and wavelength combinations to produce improved pursuit-based turning behaviour, each matching the permutations of the previous paper for comparison. For each permutation I launched 20 runs of the system, each with a population of 150 for 500 generations, chosen as the largest number of generations for a large enough population that could be executed within a reasonable runtime. Twenty was chosen as the generally accepted minimum sub-group sample size for quantitative research. I assessed how many times each walker made the target

point move away from it, which is henceforth referred to as "moves". Walkers were terminated after 60s. In order to demonstrate notable results in this chapter, I utilise the Mann-Whitney U test for significance. This was chosen as the tests all featured the effects of a single change to the system on one continuous outcome value. As there were several tests performed, the Bonferroni correction was applied as a simple way to ensure there was little room for statistical error within the tests.

Figures 8.1 and 8.2 show the maximum number of moves for the incremental versions of each set of 20 for each permutation of the amplitude and wavelength parameters. When varying amplitude, the main trend was a decrease in number of moves as amplitude increased. Again, I expected this as greater amplitudes required deeper turns. No points on the figures were below 3 moves (two turns), demonstrating the system's ability to frequently evolve two-turn behaviour. When varying wavelength, there were no standout trends. The amplitude 1 runs decreased in moves as their wavelength size increased. Compared to the standard runs, the incremental runs were similar, with the only difference being the lowest values all being above 2 for the incremental perturbations. Figures 8.3 and 8.4 show the mean number of moves for the incremental versions of each set of 20 for each permutation of the amplitude and wavelength parameters. When varying amplitude, there were no real trends besides some higher values at amplitude 1. Most of the mean values remained between 1 and 2 moves. When varying wavelength, there was a very slight upward trend toward the higher wavelengths. Compared to the standard runs, the incremental values were clustered slightly closer together by amplitude and slightly further apart by wavelength.

Figures 8.5 and 8.6 show the difference between the incremental and standard runs for each permutation, for both maximum and mean values. Comparing maximum values of each set of 20, 8 of the 25 incremental permutations had a higher maximum value than the standard. Only 2 of the permutations had a lower value, which were both in the higher amplitudes. The lowest amplitude exhibited the greatest difference, presumably as it had the highest number of moves to start with. All the amplitudes apart from 5 had at least one wavelength that had 2 more maximum moves than the standard. Several of these were in the smaller wavelengths. Comparing mean values of each set of 20, 16 of the 25 permutations had a higher mean value, with 7 permutations having a lower value. Unlike the maximum values, the means had no patterns amongst the higher or lower perturbations. Comparing the sets of runs directly, I

applied a Bonferroni correction of 25 comparisons, one for each incremental vs standard perturbation comparison. This narrowed the significance margin down to 0.002. None of the 25 incremental permutations were found to be notably higher than the standard version according to a statistical significance test. The a1w4 permutation, despite having the largest difference in mean values, only featured a p-value of 0.09342 (Mann-Whitney U=151.5, n1=n2=20, p=0.09342, one-tailed, Bonferroni correction for 25 comparisons). I also compared the gaits of the a1w4 permutations. Figures 7.7 and 8.8 show the two gaits for the a1w4 and ia1w4 permutations. The incremental version appeared to turn toward the target point slightly more, and remained facing forwards instead of veering off at the end of its lifetime. Both travelled forwards, moving the target point multiple times.

In Figure 8.7, I analysed the paths taken by the best of each of the 20 incremental walker perturbations at the 16 wavelength, as in the previous Chapter. The best agent evolved for the smallest amplitude, 1, was able to turn six times before turning back on itself. The best agent evolved with an amplitude value of 2 also had a decent fitness, managing four turns in a smooth pattern. The best agent evolved with an amplitude value of 3 stuttered briefly on its first turn but recovered, demonstrating robustness. The best agent evolved with an amplitude value of 4 and the best agent evolved with an amplitude value of 5 made two turns before veering off. Compared to the standard runs in Figure 7.6, the higher amplitude paths appeared to exhibit a much wider turning arc.

8.4 Contributions

The contributions made by this work are as follows:

- I evolved walkers incrementally on my pursuit-based turning task from Chapter 7. I tested the same set of amplitude and wavelength permutations to determine if the system could still produce multi-turn behaviours. It was still capable of this, but did not produce higher fitness, and matched neither my Chapter aim to bridge the gap between the two behaviours, or my general aim to demonstrate the benefits of evolutionary systems such as incremental evolution.

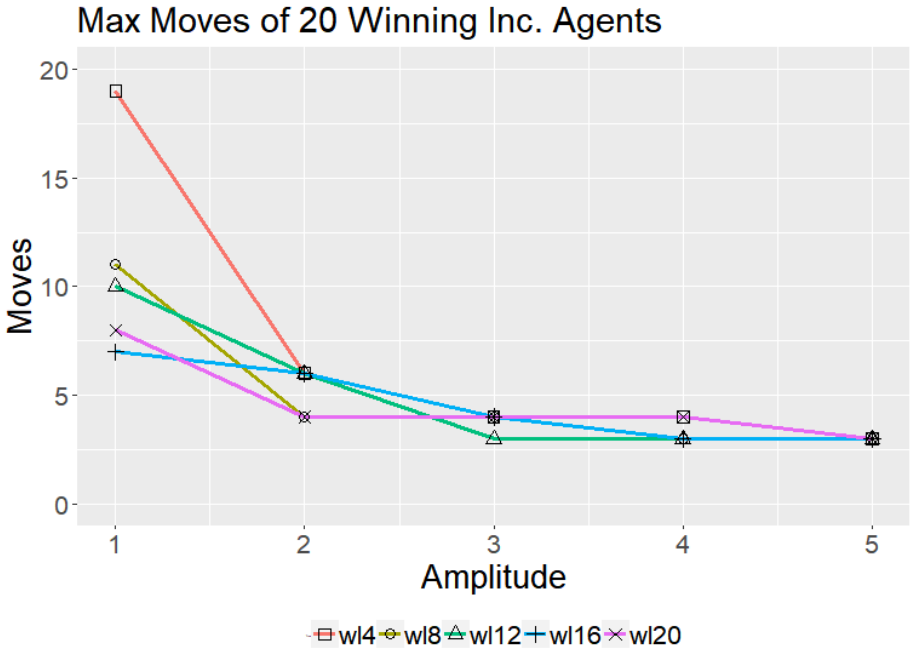

Figure 8.1: **Maximum Moves per amplitude. (Incremental)** The maximum number of moves for the incremental runs for each set of 20 agents for each wavelength parameter, at amplitudes 1,2,3,4 and 5. The maximum values decreased as amplitude increased.

8.5 Discussion

Despite the lack of notable improvements, much like in the previous Chapter, there was little distance in moves amongst the higher amplitudes and wavelengths. Figures 8.1 and 8.3 both display a decrease in range as amplitude is increased, much like those in the previous Chapter. Ranges decrease from around 5 to 0.5 on the mean values, and 10 to 1 on the maximum values. This time the trend is also more evident in Figure 8.4, with range decreasing to around 2 for the top three wavelength values. Finally, Figure 8.7 shows two clear different behaviour patterns between the high and low amplitudes. The lower amplitudes exhibit a shallow zigzagging behaviour, whilst the higher amplitudes favour a single sharp turn, similarly to the gaits seen in the previous Chapter.

There are a number of interpretations I can make, similarly to the ones in the previous Chapter, now with further context. The first notable interpretation is the difference in behaviour. As described previously, the similarities amongst higher amplitudes could imply two separate ideal behaviours for higher and lower amplitudes and wavelengths.

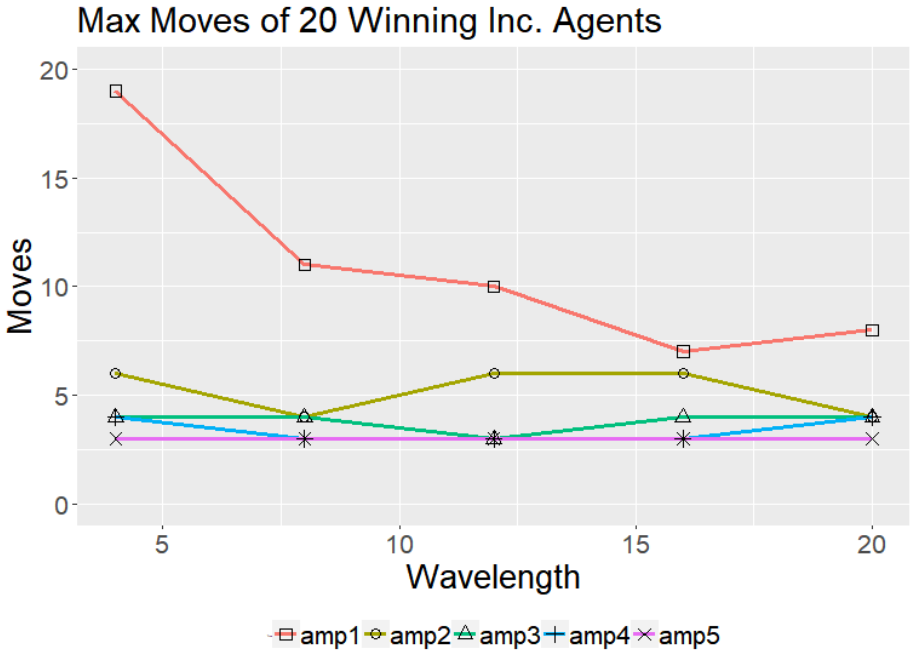

Figure 8.2: **Maximum Moves per Wavelength. (Incremental)** The maximum number of moves for the incremental runs for each set of 20 agents for each amplitude parameter, at wavelengths 4,8,12,16 and 20. The amplitude 1 runs are consistently fitter than the rest at all wavelengths.

Higher amplitudes tend to produce sharp turn behaviours, whilst lower amplitudes produce zigzagging behaviours. This is due to the difference in energy required for each behaviour; a shallow turn can be achieved by simply leaning, whilst a larger turn requires more turning and stepping motions. In addition, these trends still manifested with an incremental amplitude, potentially demonstrating that they are much more connected to the task than previously thought.

Another interpretation that can be made, particularly in light of the incremental results is that the difference between the two styles of behaviour can not be achieved via several smaller increments. This work set out to demonstrate improvements in moves using a linear incremental amplitude, but instead the results not only produced the same levels of moves, but also the qualitative split between the upper and lower amplitudes demonstrated further a considerable difference in behaviour. Instead, I could make the conclusion from this that the turning task would be better evolved in two distinct increments instead - a jump between lower amplitude leaning behaviours, and the higher amplitude stepping behaviours. After evolving the lower amplitude behaviours, the second phase between the two could then feature a modified fitness

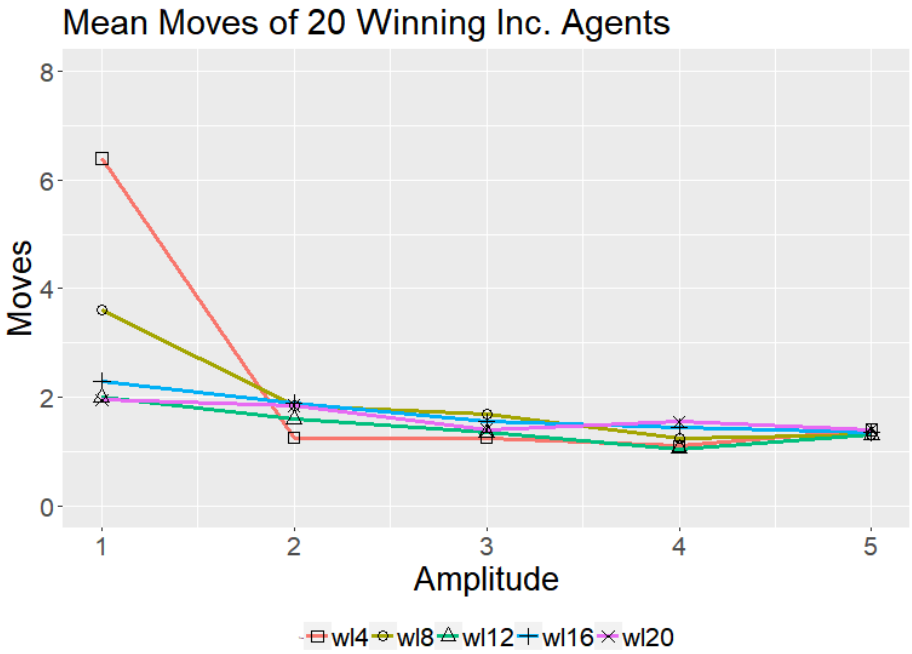

Figure 8.3: **Mean Moves per amplitude. (Incremental)** The mean number of moves for the incremental runs for each set of 20 agents for each wavelength parameter, at amplitudes 1,2,3,4 and 5. There are no real trends besides higher mean values at amplitude 1. The values are slightly more clustered than the standard mean values by amplitude.

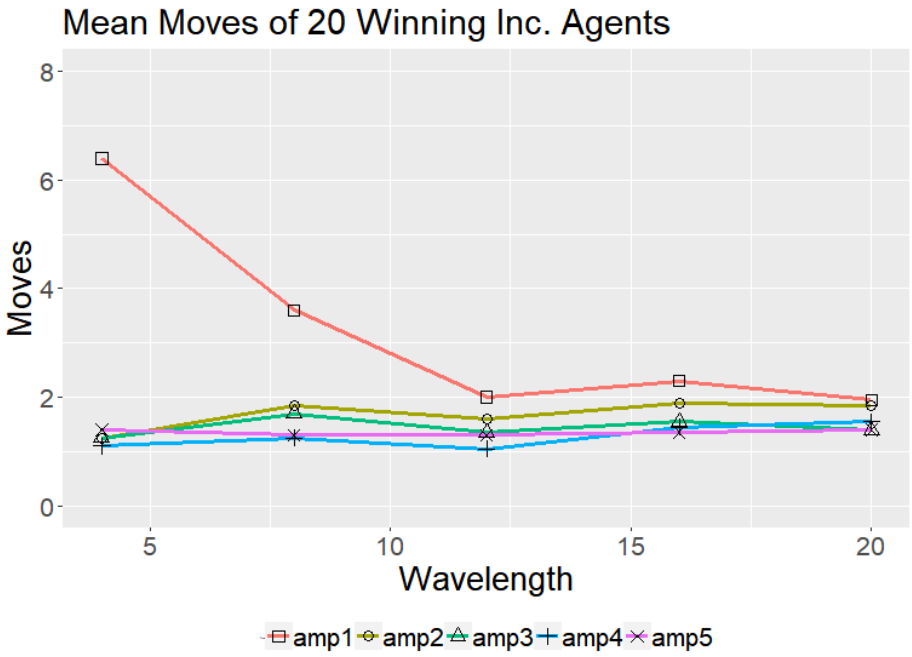

Figure 8.4: **Mean Moves per Wavelength. (Incremental)** The mean number of moves for the incremental runs for each set of 20 agents for each amplitude parameter, at wavelengths 4,8,12,16 and 20. With the exception of amplitude 1, there is a slight increase in mean values as wavelength increases. The values are slightly less clustered than the standard mean values by wavelength.

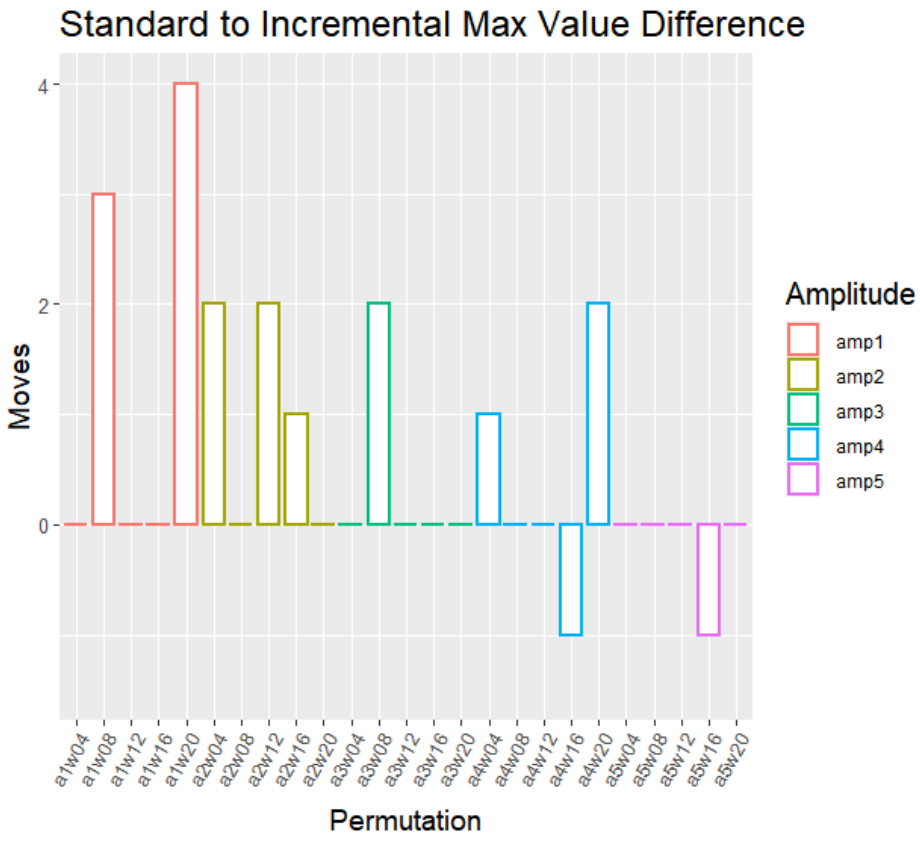

Figure 8.5: **The difference in maximum values from standard to incremental.** The difference in the maximum number of moves from standard to incremental for each set of 20 runs for each permutation, coloured by amplitude. Eight permutations had a higher value for the incremental version, whereas only two had a lower value. The amplitude 1 runs exhibited the largest differences.

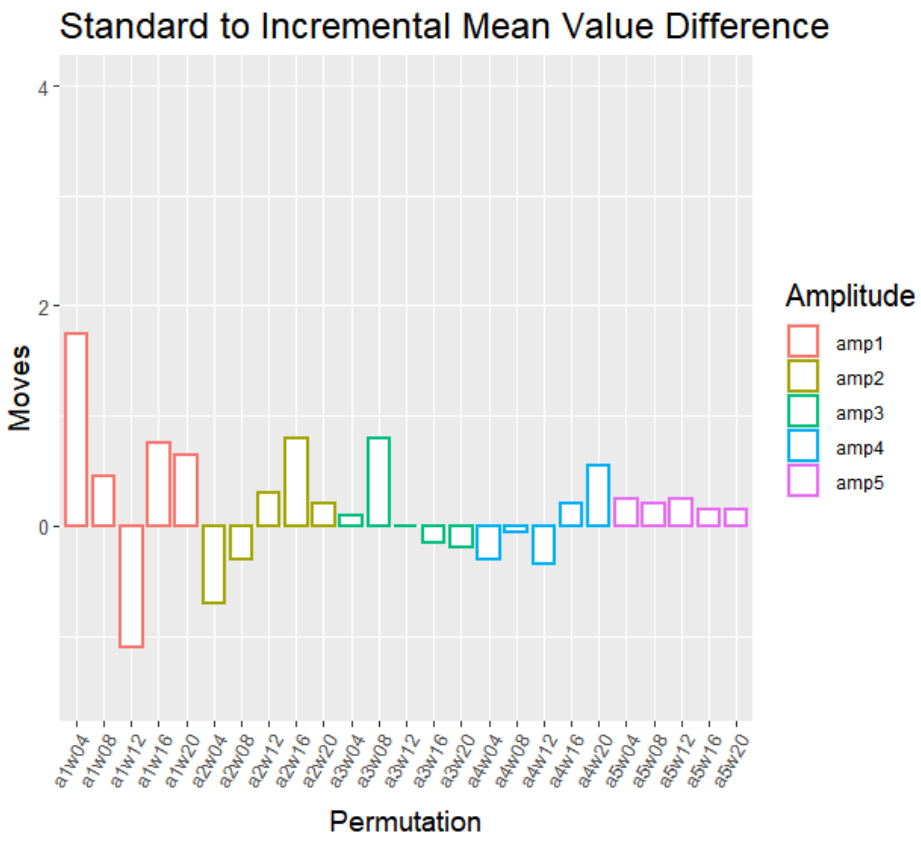

Figure 8.6: **The difference in mean values from standard to incremental.** The difference in the mean number of moves from standard to incremental for each set of 20 runs for each permutation, coloured by amplitude. Sixteen of the permutations had a larger value for the incremental version, with seven having a lower value.

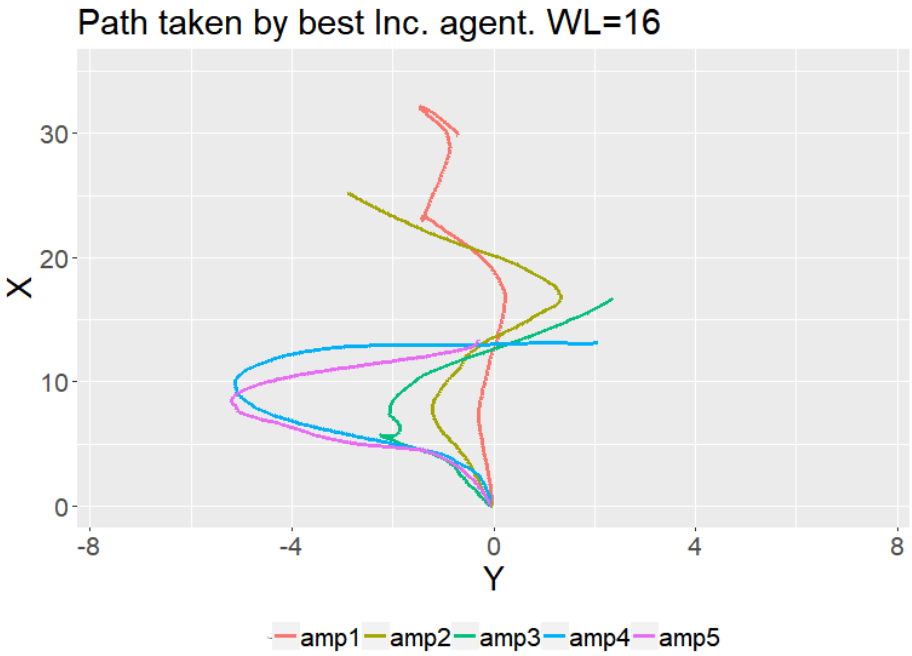

Figure 8.7: **The highest fitness incremental gaits at the 16 wavelength.** The highest fitness gait from each incremental amplitude's set of 20 incremental runs, with a wavelength parameter of 16. Each line represents the walkers path viewed from above, starting from 0,0 and following the target point in a zigzag motion. Amplitude 1 was able to turn six times. amplitude 2 managed four turns in a smooth pattern. amplitudes 3,4 and 5 each made two turns, with 3 recovering from a stumble. Compared to the defaults, the incremental runs exhibited wider turning arcs and travelled slightly further forward.

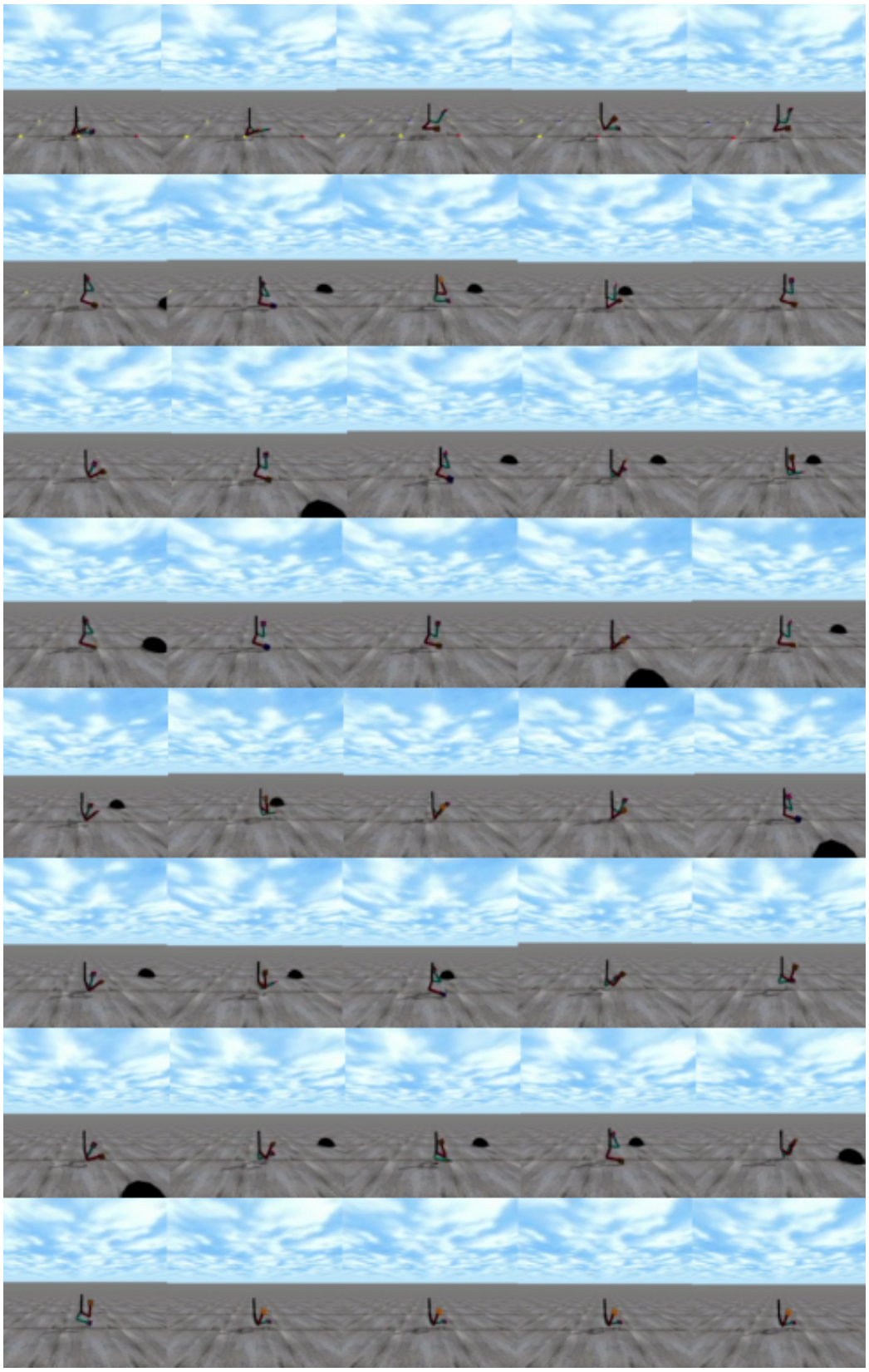

Figure 8.8: **(left to right, top to bottom) The highest fitness gait for the ia1w4 incremental perturbation.**

function or task in order to push the walkers in the ideal evolutionary direction, for improved results at higher amplitudes.

Stanton *et al* tested two different styles of incremental strategies in their work. Homogeneous strategies consisted of static and linear functions of complexity, whilst heterogeneous strategies featured oscillatory and non-linear functions. They were able to show that even the weakest heterogeneous strategy, randomly increasing the height of the wall, was able to produce better results than the best homogeneous strategy. As mentioned above, here I use a linearly scaling strategy, which they class as homogeneous. This interpretation could also suggest that investigating heterogeneous strategies for this task, particularly the oscillating heterogeneous strategies, from which Stanton *et al* recieved the best results, could see improvements in both fitness and behaviour.

8.5.1 Acknowledgements

My code is based on the work of Solomon *et al*. It is available via GitHub at *https://github.com/benjack795/solomon-reimp* for exact specification and future reproductions.

Chapter 9

Conclusions

9.1 Aims

Below I demonstrate each of my aims and how I achieved them over the course of the thesis.

9.1.1 Increase Walker Fitness.

The first aim I set at the start of the thesis was to increase walker fitness. Viewing this as a baseline for improving walker behaviour and robustness prior to more advanced tasks, I sought to enhance two bipedal systems, the Solomon *et al* Linear Reactive system, and subsequently the Salimans *et al* Deep Neuroevolution system. In Chapter 4, by adjusting the penalty for control cost I produced walkers capable of walking in 2D and 3D without the prior bootstrapping required by the original system. In Chapter 5 I utilised similar control cost reduction changes to increase agent fitness in the deep neuroevolution system, demonstrating robustness. In Chapter 6 I produced higher fitness walker agents through morphology based on a baby albatross according to statistical significance tests. The results here demonstrate that prioritising novelty over efficiency can lead to fitness and robustness improvements in bipedal walking.

9.1.2 Utilise Morphology to Simplify Bipedal Task.

The second aim I set at the start of the thesis was to simplify the bipedal walking task. Many other systems have benefited from morphology, allowing them to utilise it to complete tasks without the need for additional control complexity. I aimed to achieve similar results for bipedal walking, to pave the way for complex behaviour. In Chapters 6, 7 and 8 I utilised a morphology based on a baby albatross to simplify the bipedal walking task. The results here demonstrate that by designating parts of

the bipedal walking task to morphology instead of controller, more attention can be given to increasing fitness.

9.1.3 Produce Complex Behaviour.

Many bipedal systems have been produced, but very few featuring evolutionary processes are able to produce behaviour beyond walking forwards. Advances in the simulation of bipedal agents could serve many applications, from bipedal robotics for use in unsafe testing environments, to producing aids for people unable to walk. I stepped closer to this here. In Chapter 4 I produce single turning behaviour in the Solomon *et al* system with a distance-based fitness function. In Chapter 5 I also produce gaits with swing knees and pumping arms that appear more life-like than those produced by the original Salimans *et al* system using a control cost penalty reduction and a support polygon-based failure condition. In Chapters 7 and 8 I produce a pursuit-turning behaviour that resembles the slalom behaviour seen from Heess *et al*, with a new fitness function. The results here demonstrate that successful fitness function adjustments can produce complex novel behaviour in the bipedal walking task.

9.1.4 Highlight the merits of evolutionary processes for bipedal walking.

Evolutionary methods have some perks when compared to what I see as the current leading method for the bipedal task, the Heess *et al* reinforcement learning system. I aimed to bring more attention to evolutionary methods, showing off their perks, such as incremental evolution and biological insight. In Chapter 8 I investigated incremental evolution for the pursuit-turning task, but was unsuccessful in improving fitness. Despite this, I was able to demonstrate a clear difference in behaviour between higher and lower amplitude walkers, and therefore theorise that a two-stage incremental approach might be more applicable than the linear scaling function used in Chapter 8.

9.2 Key Techniques

9.2.1 Support Polygon

The support polygon methodology is often used for balance in robotics [86]. The method involves drawing a polygon using the points of contact between an agent and

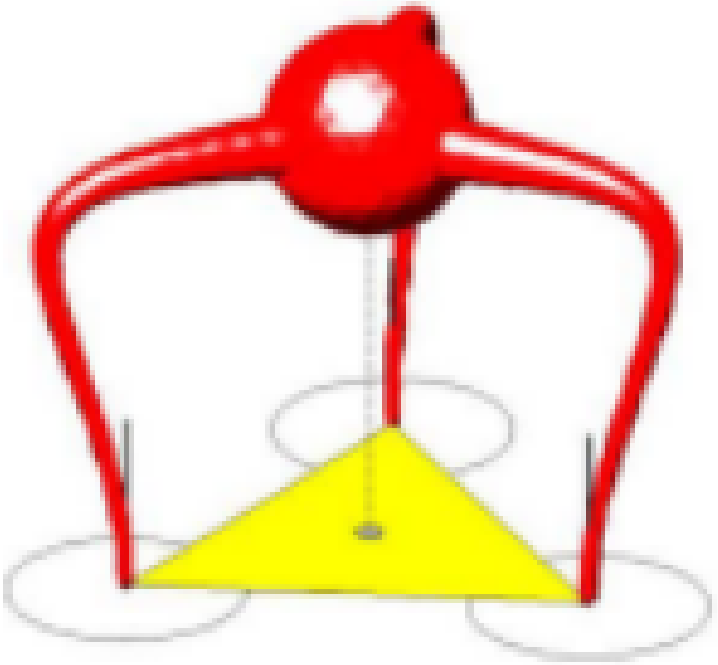

Figure 9.1: **Support Polygon Example.** An example of the support region drawn in yellow, with points of ground contact and projected centre of mass [10]. As long as the walkers projection of its centre of mass remained within the support region as it moved, it was defined as upright. I applied this to my deep walkers in Chapter 5, producing a support circle from the positions of the walker's two feet.

Figure 9.2: **The Baby Albatross** The baby albatross is seen here with its soft feathered underbelly which it uses to cushion falls and roll upright [49]. I used this underbelly as inspiration for my *square-base* and *big-tucked* morphologies in Chapter 6, giving them sphere attachments to help them roll and self-right when they fell.

the ground, projected downwards in the z-axis. From there, the centre of mass is then projected downwards onto the floor, also in the z-axis. If this projection falls within the polygon (the dubbed "support polygon"), then the agent is classed as balanced as its centre of mass has not deviated to far from its points of contact. This allows for a non-intensive framework for determining whether an agent will topple. This method is also convenient, as it can apply to any number of contact points with the ground; in my case, each of the walkers' two feet. I chose this method as it allowed me to add a balance-enforcing fitness modification to the system, demonstrating the effectiveness of fitness modifications for the bipedal walking task. I applied the technique to the deep walking system in Chapter 5 as an additional fail condition, allowing me to produce more robust walking behaviours and a gait pumping its arm for stability. In addition, the distributions of the combination runs in Chapter 5 closely resembled the support polygon runs' distributions, suggesting that this was the more important of the two techniques in the combination. A similar balance-focused method was employed in Chapter 6, in which a square-based morphology enabled walkers to remain upright.

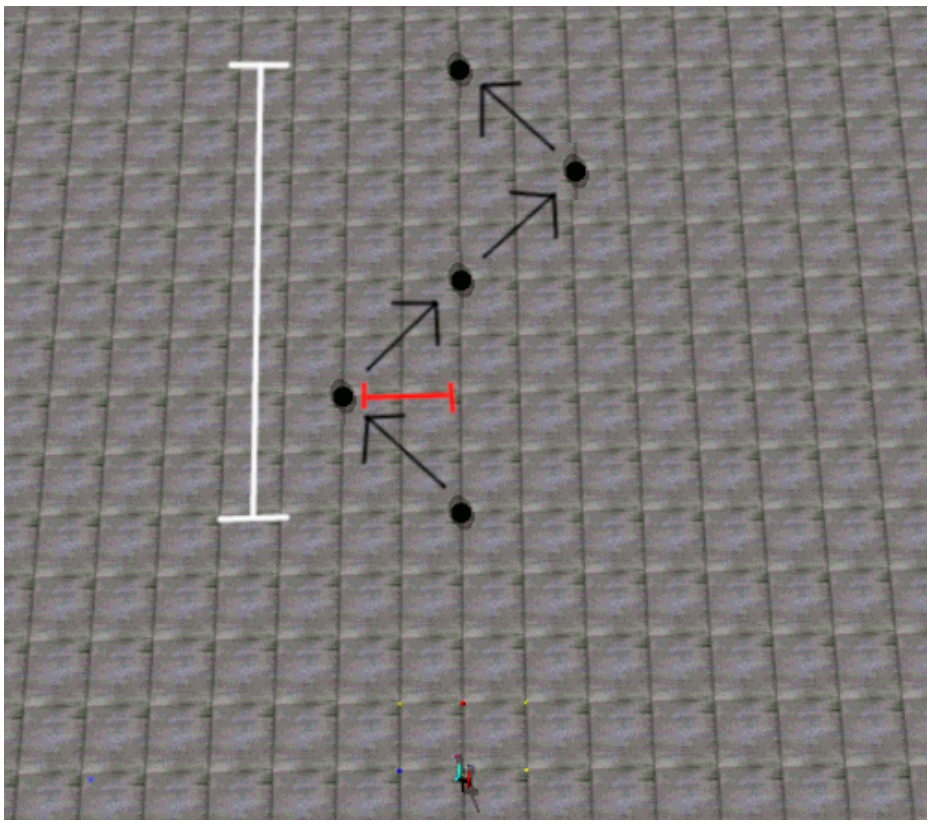

Figure 9.3: **My Turning Task.** The walkers in Chapters 7 and 8 (bottom-middle) had to follow the black target point, starting from a little in front of them, as it moved back and forth across the x-axis and away, forcing them to turn back and forth. I varied the amplitude (red) of the pursuit pattern incrementally in Chapter 8, linearly increasing it from 0 to a maximum value at half generations. The maximum was then maintained for the latter half of generations.

9.2.2 Albatross Morphology

The albatross has a feathered underbelly which cushions it from falls and allows it to right itself by rolling if it falls from its nest. Feathers have been demonstrated to assist with complex tasks by both Shim *et al* and Chang *et al* [72] [15]. The albatross also has elongated ankles, like most birds, allowing its legs to hoist up the body like a scissor lift. These two aspects of morphology allow for the bird to self-right and realign itself, which is a useful trait for robust bipedal walking. I chose this technique as an example of morphology in nature proven to assist with complex tasks, in order to demonstrate the effectiveness of morphological assistance for the bipedal walking task. In Chapter 6, I applied these features via spherical support mechanisms and inverted knee joints. The spherical support mechanisms were attached to walker bodies and exempt from the system's fall condition. The systems fall condition stated that any part of the body above the knee touching the floor was classed as a fall. The sphere attachments were exempt from this, allowing the walker to roll over on them to right itself. Whilst this prevented them from falling at all due to the size of the spheres, it did not guarantee fitness. A walker could flail endlessly without moving forwards. I gave the inverted knee joints degrees of freedom to match stereotypical bird legs, allowing the walker to lift itself up after righting. These modifications allowed the walkers to dramatically increase their fitness, travelling much further now that their morphology contributed toward their behaviour. I would later employ this again in Chapters 7 and 8, using the robustness the morphology provided to evolve multiple-turn behaviour.

9.2.3 Incremental Evolution

Incremental evolution involves breaking a complex task down into sub-tasks, and evolving through them one at a time. In this way, evolution has a smoother path to follow than through attempting to learn the entire task at once. It can also be useful to avoid "catastrophic forgetting", the idea of forgetting one task when learning another. It was used by Solomon *et al* to bootstrap their neural network controllers for the Linear Reactive system. It was also used by Stanton *et al* to enable quadrupedal agents to climb over walls of varying heights [79]. I chose this technique to demonstrate the benefits of producing bipedal walking with evolutionary processes instead of machine learning, as it is unique to evolutionary processes. In Chapter 8, after noting Solomon *et al* incrementally evolving between controller schemes, Stanton *et al* incrementally evolving between different heights of a wall for a climbing task,

and my conclusion that the task produced two separate classes of gait from Chapter 7, I investigated an incremental version of the pursuit-turning task from Chapter 7, linearly scaling the amplitude (left/right) value of the target point per generation, up to the maximum value at half the total number of generations. Despite aiming to highlight the benefits of evolutionary processes, this did not produce improved fitness.

9.3 Contributions

The contributions for the thesis across each results Chapter are explicitly defined as follows:

9.3.1 Chapter 4

- **Produced an enhanced version of the Solomon *et al* bipedal system directly into 3D without prior bootstrapping for future use.** By introducing control enhancements to the Solomon *et al* Linear Reactive system (removal of torque limit after a given time, enabling a flight phase, replacement of PD controls with ODE methods), alongside an additional 15% elitism, I demonstrated they enabled the evolution of fully 3D walking behaviours within the system, without any prior 2D bootstrapping [77]. The enhanced system they produce is simple enough through the Linear Reactive controller network and the lack of bootstrapping required that it can serve as a basis for future bipedal works. This matches my Chapter aim to remove bootstrapping and my general aim to increase walker fitness.

- **Produced 2D walking gaits in the enhanced system matching the original.** I demonstrate the control enhancements enable the production of successful life-like 2D constrained gaits in the enhanced system, taking as many as 30 steps and matching the original system in fitness.

- **Produced 3D walking gaits in the enhanced system.** I demonstrate the control enhancements enable the production of 3D gaits capable of as many as 8 steps in 3D in the enhanced system, although these were not as successful as the original system.

- **Produced singular lateral turning behaviour in the enhanced system.** I demonstrate the control enhancements enable the production of single left-right turning behaviour in the enhanced system by applying a new distance-

based fitness function. This behaviour was comparable to that seen in Reil *et al* [64]. This matches my Chapter aim and general aim to produce complex behaviour.

9.3.2 Chapter 5

- **Improved fitness in the Salimans *et al* bipedal system by introducing enhancements to the fitness function.** I reduce control cost within the fitness function, based upon an adjustment to the torque limit in the Solomon *et al* system in the previous Chapter, prioritising novelty over efficiency. When control cost was reduced to a quarter of the default, from the 500th time-step (the halfway point for a full-length episode), median speed and median distance both doubled. I also terminate walking when the torso's centre of mass moves outside a circular support polygon centered at the midpoint between the walker's feet, based around an existing centre of mass condition within the system and the importance of balance to bipedal tasks overall. When the circle's radius was 0.75 times current distance between the midpoint and (either) foot, median speed and median distance are both found to be higher according to a statistical significance test. This demonstrates robustness of these methods in a new system. This matches my Chapter aim to test the robustness of my methods and my general aim to increase walker fitness.

- **Produced gaits with the enhancements that overcome the shuffling problem seen in several gaits produced from the Salimans *et al* system.** I produce notable gaits using the enhancements with a more pronounced stance and swing phase. One puts one leg in front of the other in a clearer fashion than any gait produced using the default fitness function, getting around the issue of shuffling behaviours. The other produces an arm pumping motion similar to arm swinging in humans. This demonstrates my results from these techniques approaches those produced by machine learning. This matches my Chapter and general aim to produce complex behaviour.

- **Produced walkers robust to action noise in the Salimans *et al* system by combining the control cost enhancement and support polygon constraint together.** I evaluated evolved walkers with the addition of noise to their actions. The combination runs do not drop off in fitness in the presence of low noise as quickly as the other types. In contrast to the without-noise results, the combined-enhancement gaits showed large improvements over those

from the individual control-cost and balance enhancements, with median speed, distance and time higher than the default and individual-enhancement gaits at intermediate levels of noise according to a statistical significance test. This demonstrates the combined enhancement drops off slower in the presence of low noise and matches my general aim to increase walker fitness. The distribution of the combo runs also more closely resembles the that of the rp75 than the s500. This suggests that the rp75 has more impact than the s500, and potentially therefore that balance is more important than efficiency or novelty for robustness.

9.3.3 Chapter 6

- **Produced fitter bipedal walkers in the enhanced Solomon *et al* system with a square-based morphology based on a baby albatross.** I evolved bipedal walking agents alongside a square-based morphology based on the underbelly of a baby albatross, following on from the support polygon balance enforcement rationale of the previous Chapter. The simpler neural network controller in the ODE system was chosen to be more sensitive to morphological changes and require less run-time. This demonstrates this morphology produces agents fitter than the default according to statistical significance tests. This matches my Chapter aim to use the square-based morphology to increase fitness and my general aim to utilise morphology to simplify the bipedal task.

- **Produced fitter bipedal walkers in the enhanced Solomon *et al* system with a tucked-knees morphology based on a baby albatross.** I also evolved bipedal walking agents alongside a morphology based on the underbelly of a baby albatross featuring tucked knees, following rationale of more motion from the previous Chapter. The simpler neural network controller in the ODE system was chosen to be more sensitive to morphological changes and require less run-time. This demonstrates this morphology produces agents fitter than the default according to statistical significance tests. This matches my Chapter aim to use the tucked-knees morphology to increase fitness and my general aim to utilise morphology to simplify the bipedal task.

- **Investigated evolving morphology alongside control using the albatross morphologies, for both randomly initialised and incrementally preloaded genotypes.** I evolved these morphologies alongside the neural network controller, creating a second morphological genotype for leg length and

seeding both with previously high-fitness agents. Randomly initialising these morphology and neural network controller genotypes instead achieved fitness up to four times higher. Whilst this does not achieve my Chapter aim of returning to standard proportions with increased fitness, this is still in line with my general aim to utilise morphology to simplify the bipedal task.

- **Produced a fast gait with a randomly-initialised morphology genotype which featured higher input weights for important neural network controllers.** I also evolved a *big-tucked* randomly-initialised morphology agent with higher input weights for its upper body and swing knee neural network controllers according to a statistical significance test. This agent featured an extremely fast gait as a result. This matches my aim to increase walker fitness.

9.3.4 Chapter 7

- **Produced albatross morphology bipedal walkers capable of novel behaviour, turning multiple times whilst pursuing a moving point.** I evolved walkers in my enhanced re-implementation of the Solomon *et al* Linear Reactive system from Chapter 4, together with my high-fitness *big-tucked* albatross morphology from Chapter 6, on a pursuit-based turning task, designed as an extension of the single-turn behaviours produced in Chapter 4. This was chosen instead of the Salimans *et al* system to further demonstrate the utility of morphological changes by retaining them alongside complex behaviour. Explicitly, this demonstrates the pursuit-turning task produces walkers capable of producing four consecutive turns, and two turns at high amplitudes, in pursuit of a moving point. The pursuit-based turning behaviour evolved here is also an example of complex behaviour that, to the best of my knowledge, has not been attained previously in evolved bipedal walkers. This was difficult previously as traditional bipedal agents have a high center of mass and feet close together, making them tricky to balance alongside other complex tasks. The behaviour resembles the the slalom behaviour produced by Heess *et al*. This matches my Chapter aim to produce pursuit-turning behaviour and general aim to produce complex behaviour.

9.3.5 Chapter 8

- **Investigated incrementally evolving pursuit-turning bipedal walkers.** I evolved walkers incrementally on my pursuit-based turning task from Chapter

7. I tested the same set of amplitude and wavelength permutations to determine if the system could still produce multi-turn behaviours. It was still capable of this, but did not produce higher fitness, and matched neither my Chapter aim to bridge the gap between the two behaviours, or my general aim to demonstrate the benefits of evolutionary systems such as incremental evolution.

9.4 Further Work and Discussion

Further works could focus more on the novelty over efficiency rationale demonstrated across several Chapters here. In Chapter 4, the energy cost failure condition was removed after a given period of time to allow for more motion, leading to a system that could evolve 3D walking behaviours without any prior bootstrapping. In Chapter 5, a similar reduction in energy cost was applied to the fitness function of the system to allow for more motion, allowing for improved fitness and the emergence of a gait that put one foot in front of the other, as opposed to simply shuffling. Finally in Chapter 6, the *big-tucked* morphology based on the baby albatross allowed for more movement with its tucked knees and the energy cost constraint removed, which produced fitter walker agents according to a statistical significance test. This can be interpreted as evidence that sacrificing efficiency for novelty is a useful technique to consider when aiming for complex behaviour. Future works could look at simultaneously adding novelty methodologies, such as novelty search whilst reducing the impact of efficiency penalties to other successful bipedal systems in order to further demonstrate the robustness of this technique [43]. They could also utilise this philosophy to produce further behaviours, weighted alongside other fitness enhancements, similar to those seen in Heess *et al*, such as running or hurdle jumping [35].

Further works could also focus more on fitness function enhancements and their ability to influence an evolutionary system more than other types of changes. In Chapter 4, in order to produce left-right turning behaviour, a new fitness function based around distance was applied, producing successful single turns. In Chapter 5, the control cost penalty term of the fitness function is modified, again producing improved fitness and a gait putting one foot in front of the other instead of shuffling. In Chapter 7, in order to produce slalom turning behaviour matching Heess *et al*, a new fitness function again based around distance was applied, evolving behaviour making as many as four turns at lower values. This can be interpreted as evidence that modifying fitness functions is the best way to influence an evolutionary system and the behaviour it

produces. Future works could look at alternate fitness functions for existing bipedal systems to produce novel behaviours not previously seen. This interpretation could also be applied to future evolved bipedal control tasks, designing them around key fitness functions, potentially leading to the emergence of more complex behaviour in bipeds.

Another potential avenue for further work is to provide additional motivation for the selection of evolutionary methods over other methods such as reinforcement learning. In Chapter 5, I produce increased fitness and complex gaits through modification of a fitness function in a system that was already capable of matching reinforcement learning at the humanoid walking task. In Chapter 7 I produce an as-of-yet unseen behaviour through a novel fitness function, and in Chapter 8 whilst I do not produce any fitness increase, I suggest an incremental evolution based method to bridge the gap between the two separate walking behaviours that emerged across both versions of the pursuit-turning task. These results can be interpreted as demonstrating the viability of evolutionary systems as a whole for developing complex bipedal behaviour. This could lead to future works analysing the evolutionary trajectory provided through the evolution of bipedal behaviour to compare it with that of ours and other species within the biosphere, to uncover insights not produced from other methods.

A final extension of this work would be to achieve a further complex task incrementally by using the existing turning task as a sub-task. In Chapter 7, I achieved the ability to turn across the x-axis, and noted the difference between the behaviours produced at low and high amplitudes, almost as two separate turning tasks. Later in Chapter 8, I investigated incrementally scaling between the two with a linear amplitude function, but this was not fitter. However, this still provides a useful interpretation. In Chapter 8 I note that whilst this was not a success, the vast difference in behaviour between high and low amplitudes could perhaps be better split as two separate phases to bootstrap between instead of scaling linearly. If this split were successful, I could use the higher of the two turning levels as a starting point to incrementally evolve random turning pursuit, or even avoiding moving obstacles. This would test the walkers' ability to use their sensors, evolving responsive motion. Responsive behaviour in relation to the environment would represent a much more intelligent, situated agent.

References

[1] Brian F. Allen and Petros Faloutsos. Evolved controllers for simulated locomotion. In *International Workshop on Motion in Games*, 2009.

[2] Amin Azarbadegan, Frank Broz, and Chrystopher L. Nehaniv. Evolving sims's creatures for bipedal gait. In *2011 IEEE Symposium on Artificial Life (ALIFE)*, 2011.

[3] Lionel Barnett. *Evolutionary Search on Fitness Landscapes with Neutral Networks*. PhD thesis, University of Sussex, 2003.

[4] Sven Behnke. Online trajectory generation for omnidirectional biped walking. In *Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006.*, 2006.

[5] Asgeir Berland. A cheap spiking neural model for evolved controllers. Master's thesis, Utrecht University, 2016.

[6] Gaurav Bharaj, Stelian Coros, Bernhard Thomaszewski, James Tompkin, Bernd Bickel, and Hanspeter Pfister. Computational design of walking automata. In *Proceedings of the 14th ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, 2015.

[7] Adrian Boeing and Thomas Bräunl. Evaluation of real-time physics simulation systems. In *Proceedings of the 5th international conference on Computer graphics and interactive techniques in Australia and Southeast Asia*, 2007.

[8] Josh Bongard. The utility of evolving simulated robot morphology increases with task complexity for object manipulation. *Artificial Life*, 16, 2010.

[9] Dennis M. Bramble and Daniel E. Lieberman. Endurance running and the evolution of homo. *Nature*, 432, 2004.

[10] Timothy Bretl and Sanjay Lall. Testing static equilibrium for legged robots. *IEEE Transactions on Robotics*, 24, 2008.

[11] Adrien Briod, Adam Klaptocz, Jean-Christophe Zufferey, and Dario Floreano. The airburr: A flying robot that can exploit collisions. In *2012 ICME International Conference on Complex Medical Engineering (CME)*, 2012.

[12] Rodney A. Brooks. Intelligence without representation. *Artificial Intelligence*, 47, 1991.

[13] Marcello Calisti, Michele Giorelli, Guy Levy, Barbara Mazzolai, B. Hochner, Cecilia Laschi, and Paolo Dario. An octopus-bioinspired solution to movement and manipulation for soft robots. *Bioinspiration & Biomimetics*, 6, 2011.

[14] Jorge Cervantes and Christopher R. Stephens. Limitations of existing mutation rate heuristics and how a rank ga overcomes them. *IEEE Transactions on Evolutionary Computation*, 13, 2008.

[15] Eric Chang, Laura Y. Matloff, Amanda K. Stowers, and David Lentink. Soft biohybrid morphing wings with feathers underactuated by wrist and finger motion. *Science Robotics*, 5, 2020.

[16] Nicolas Chaumont, Richard Egli, and Christoph Adami. Evolving virtual creatures and catapults. *Artificial Life*, 13, 2007.

[17] David T. Cli, Inman Harvey, and Philip Husbands. Incremental evolution of neural network architectures for adaptive behaviour. In *Proceedings of the European Symposium on Artificial Neural Networks (ESANN'93)*, 1992.

[18] Steven H. Collins, Peter G. Adamczyk, and Arthur D. Kuo. Dynamic arm swinging in human walking. *Proceedings of the Royal Society B: Biological Sciences*, 276, 2009.

[19] Steven H. Collins, Martijn Wisse, and Andy Ruina. A three-dimensional passive-dynamic walking robot with two legs and knees. *The International Journal of Robotics Research*, 20, 2001.

[20] Edoardo Conti, Vashisht Madhavan, Felipe Petroski Such, Joel Lehman, Kenneth O. Stanley, and Jeff Clune. Improving exploration in evolution strategies for deep reinforcement learning via a population of novelty-seeking agents, 2018.

[21] Francesco Corucci, Marcello Calisti, Helmut Hauser, and Cecilia Laschi. Novelty-based evolutionary design of morphing underwater robots. In *Proceedings of the 2015 Annual Conference on Genetic and Evolutionary Computation*, 2015.

[22] Alessandro Crespi, Daisy Lachat, Ariane Pasquier, and Auke Jan Ijspeert. Controlling swimming and crawling in a fish robot using a central pattern generator. *Autonomous Robots*, 25, 2008.

[23] Antoine Cully and J.B. Mouret. Evolving a behavioral repertoire for a walking robot. *Evolutionary Computation*, 24, 2016.

[24] Charles Darwin, Mortimer Jerome Adler, and Robert Maynard Hutchins. *On the Origins of Species by Means of Natural Selection*. Encyclopaedia Britannica, 1952.

[25] Fred Delcomyn. Neural basis of rhythmic behavior in animals. *Science*, 210, 1980.

[26] Jan Dyck. The evolution of feathers. *Zoologica Scripta*, 14, 1985.

[27] Boston Dynamics. Atlas. https://www.bostondynamics.com/atlas. Accessed: 2021-05-11.

[28] Paul R. Ehrlich and Peter H. Raven. Butterflies and plants: A study in coevolution. *Evolution*, 18, 1964.

[29] Tom Erez, Yuval Tassa, and Emanuel Todorov. Simulation tools for model-based robotics: Comparison of bullet, havok, mujoco, ode and physx. In *2015 IEEE international conference on robotics and automation (ICRA)*, 2015.

[30] Lawrence J. Fogel, Alvin J. Owens, and Michael J. Walsh. *Artificial intelligence through simulated evolution*. Wiley, 1966.

[31] Thomas Geijtenbeek, Michiel Van De Panne, and A. Frank Van Der Stappen. Flexible muscle-based locomotion for bipedal creatures. *ACM Transactions on Graphics (TOG)*, 32, 2013.

[32] Faustino Gomez and Risto Miikkulainen. Incremental evolution of complex general behavior. *Adaptive Behavior*, 5, 1997.

[33] John J. Grefenstette. Optimization of control parameters for genetic algorithms. *IEEE Transactions on Systems, Man, and Cybernetics*, 16, 1986.

[34] Inman Harvey. *Evolutionary Robotics and SAGA: the case for Hill Crawling and Tournament Selection*. University of Sussex, School of Cognitive and Computing Sciences, 1992.

[35] Nicolas Heess, Sriram S TB D, J Lemmon, J Merel, G Wayne, et al. Emergence of locomotion behaviours in rich environments. *Available: arxiv*, 1707, 2017.

[36] Gordon W. Hewes. Food transport and the origin of hominid bipedalism. *American Anthropologist*, 63, 1961.

[37] John H. Holland. Outline for a logical theory of adaptive systems. *Journal of the ACM (JACM)*, 9, 1962.

[38] Pei-Chi Huang, Joel Lehman, Aloysius K. Mok, Risto Miikkulainen, and Luis Sentis. Grasping novel objects with a dexterous robotic hand through neuroevolution. In *2014 IEEE Symposium on Computational Intelligence in Control and Automation (CICA)*, 2014.

[39] Auke Jan Ijspeert, Alessandro Crespi, Dimitri Ryczko, and Jean-Marie Cabelguen. From swimming to walking with a salamander robot driven by a spinal cord model. *Science*, 315, 2007.

[40] Ben Jolley and Alastair Channon. Evolving robust, deliberate motion planning with a shallow convolutional neural network. In *Artificial Life Conference Proceedings*, 2018.

[41] Fumio Kanehiro, Masayuki Inaba, and Hirochika Inoue. Development of a two-armed bipedal robot that can walk and carry objects. In *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems. IROS'96*, 1996.

[42] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. *Advances in Neural Information Processing Systems*, 25, 2012.

[43] Joel Lehman and Kenneth O. Stanley. Abandoning objectives: Evolution through the search for novelty alone. *Evolutionary Computation*, 19, 2011.

[44] Dan Lessin, Don Fussell, and Risto Miikkulainen. Open-ended behavioral complexity for evolved virtual creatures. In *Proceedings of the 15th Annual Conference on Genetic and Evolutionary Computation*, 2013.

[45] Dan Lessin, Don Fussell, and Risto Miikkulainen. Adapting morphology to multiple tasks in evolved virtual creatures. In *Artificial Life Conference Proceedings 14*, 2014.

[46] Timothy P. Lillicrap, Jonathan J. Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning, 2019.

[47] Hod Lipson and Jordan B. Pollack. Automatic design and manufacture of robotic lifeforms. *Nature*, 406, 2000.

[48] Hod Lipson, Vytas Sunspiral, Josh Bongard, and Nicholas Cheney. On the difficulty of co-optimizing morphology and control in evolved virtual creatures. In *Artificial Life Conference Proceedings 13*, 2016.

[49] Rod Long. A black-browed albatross chick. https://unsplash.com/photos/IGyU-eCMW5A. Accessed: 2021-05-11.

[50] Poramate Manoonpong, Tao Geng, Bernd Porr, and Florentin Worgotter. The runbot architecture for adaptive, fast, dynamic walking. In *2007 IEEE International Symposium on Circuits and Systems*, 2007.

[51] Tad McGeer. Passive dynamic walking. *The International Journal of Robotics Research*, 9, 1990.

[52] Thomas Miconi. Evolution and complexity: The double-edged sword. *Artificial Life*, 14, 2008.

[53] Thomas Miconi. In silicon no one can hear you scream: Evolving fighting creatures. In *European Conference on Genetic Programming*, 2008.

[54] Marcell Missura and Sven Behnke. Gradient-driven online learning of bipedal push recovery. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2015.

[55] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing atari with deep reinforcement learning, 2013.

[56] Hans P. Moravec. The stanford cart and the cmu rover. *Proceedings of the IEEE*, 71, 1983.

[57] J.B. Mouret and Stéphane Doncieux. Encouraging behavioral diversity in evolutionary robotics: An empirical study. *Evolutionary Computation*, 20, 2012.

[58] Jean-Baptiste Mouret, Stéphane Doncieux, and Jean-Arcady Meyer. Incremental evolution of target-following neuro-controllers for flapping-wing animats. In *International Conference on Simulation of Adaptive Behavior*, 2006.

[59] Martin Nowak and Karl Sigmund. Chaos and the evolution of cooperation. *Proceedings of the National Academy of Sciences*, 90, 1993.

[60] Tønnes F. Nygaard, Jim Torresen, and Kyrre Glette. Multi-objective evolution of fast and stable gaits on a physical quadruped robotic platform. In *2016 IEEE Symposium Series on Computational Intelligence (SSCI)*, 2016.

[61] Gabriela Ochoa, Inman Harvey, and Hilary Buxton. Error thresholds and their relation to optimal mutation rates. In *European Conference on Artificial Life*, 1999.

[62] Chandana Paul and Josh C. Bongard. The road less travelled: Morphology in the optimization of biped robot locomotion. In *Proceedings 2001 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2001.

[63] Ingo Rechenberg. Evolutionsstrategien. In *Simulationsmethoden in der Medizin und Biologie*. Springer, 1978.

[64] Torsten Reil and Phil Husbands. Evolution of central pattern generators for bipedal walking in a real-time physics environment. *IEEE Transactions on Evolutionary Computation*, 6, 2002.

[65] Edward Robinson, Timothy Ellis, and Alastair Channon. Neuroevolution of agents capable of reactive and deliberative behaviours in novel and dynamic environments. In *European Conference on Artificial Life*, 2007.

[66] David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. Learning representations by back-propagating errors. *Nature*, 323, 1986.

[67] Yoshiaki Sakagami, Ryujin Watanabe, Chiaki Aoyama, Shinichi Matsunaga, Nobuo Higaki, and Kikuo Fujimura. The intelligent asimo: System overview and integration. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2002.

[68] Tim Salimans, Jonathan Ho, Xi Chen, Szymon Sidor, and Ilya Sutskever. Evolution strategies as a scalable alternative to reinforcement learning, 2017.

[69] Tim Salimans, Jonathan Ho, Xi Chen, and Ilya Sutskever. Distributed evolution. https://github.com/openai/evolution-strategies-starter. Accessed: 2021-05-11.

[70] Cristina P. Santos and Vítor Matos. Gait transition and modulation in a quadruped robot: A brainstem-like modulation approach. *Robotics and Autonomous Systems*, 59, 2011.

[71] José Santos and Angel Campo. Biped locomotion control with evolved adaptive center-crossing continuous time recurrent neural networks. *Neurocomputing*, 86, 2012.

[72] Yoonsik Shim and Phil Husbands. Feathered flyer: Integrating morphological computation and sensory reflexes into a physically simulated flapping-wing robot for robust flight manoeuvre. In *European Conference on Artificial Life*, 2007.

[73] Pedro Silva, Cristina P. Santos, Vítor Matos, and Lino Costa. Automatic generation of biped locomotion controllers using genetic programming. *Robotics and Autonomous Systems*, 62, 2014.

[74] Karl Sims. Evolving 3d morphology and behavior by competition. *Artificial Life*, 1, 1994.

[75] Karl Sims. Evolving virtual creatures. In *Proceedings of the 21st Annual Conference on Computer Graphics and Interactive Techniques*, 1994.

[76] Russell Smith. *Open Dynamics Engine*, 2005.

[77] Joseph H. Solomon, Mark A. Locascio, and Mitra J.Z. Hartmann. Linear reactive control for efficient 2d and 3d bipedal walking over rough terrain. *Adaptive Behavior*, 21, 2013.

[78] Kenneth O. Stanley and Risto Miikkulainen. Evolving neural networks through augmenting topologies. *Evolutionary Computation*, 10, 2002.

[79] Adam Stanton and Alastair Channon. Heterogeneous complexification strategies robustly outperform homogeneous strategies for incremental evolution. In *Artificial Life Conference Proceedings 13*, 2013.

[80] Adam Stanton and Alastair Channon. Incremental neuroevolution of reactive and deliberative 3d agents. In *Artificial Life Conference Proceedings 13*, 2015.

[81] Felipe Petroski Such, Vashisht Madhavan, Edoardo Conti, Joel Lehman, Kenneth O. Stanley, and Jeff Clune. Ai labs neuroevolution algorithms. https://github.com/uber-research/deep-neuroevolution. Accessed: 2021-05-11.

[82] Felipe Petroski Such, Vashisht Madhavan, Edoardo Conti, Joel Lehman, Kenneth O. Stanley, and Jeff Clune. Deep neuroevolution: Genetic algorithms are a competitive alternative for training deep neural networks for reinforcement learning, 2018.

[83] S. Uyar, G. Eryigit, and Sanem Sariel. An adaptive mutation scheme in genetic algorithms for fastening the convergence to the optimum. In *Proceedings of 3rd APIS: Asian Pacific International Symposium on Information Technologies*, 2004.

[84] Michiel Van de Panne and Alexis Lamouret. Guided optimization for balanced locomotion. In *Computer Animation and Simulation '95*. Springer, 1995.

[85] Eric D. Vaughan. *The evolution of an omni-directional bipedal robot*. PhD thesis, University of Sussex, 2007.

[86] Miomir Vukobratović and Branislav Borovac. Zero-moment point—thirty five years of its life. *International Journal of Humanoid Robotics*, 1, 2004.

[87] Michael Wagner, Wentong Cai, Michael H. Lees, and Heiko Aydt. Evolving agent-based models using self-adaptive complexification. *Journal of Computational Science*, 10, 2015.

[88] Christopher John Cornish Hellaby Watkins. *Learning from Delayed Rewards*. PhD thesis, King's College, Cambridge, 1989.

[89] Richard A. Watson and Jordan B. Pollack. Coevolutionary dynamics in a minimal substrate. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO 2001)*, 2001.

[90] Allen Waxman, J. Moigne, and Babu Srinivasan. Visual navigation of roadways. In *Proceedings. 1985 IEEE International Conference on Robotics and Automation*, 1985.

[91] Lukasz Wiklendt, ST Chalup, and Marıa M. Seron. Simulated 3d biped walking with an evolution-strategy tuned spiking neural network. *Neural Network World*, 19, 2009.

[92] Claus O. Wilke. Adaptive evolution on neutral networks. *Bulletin of Mathematical Biology*, 63, 2001.

[93] Krister Wolff, Jimmy Pettersson, Almir Heralic, and Mattias Wahde. Structural evolution of central pattern generators for bipedal walking in 3d simulation. In *2006 IEEE International Conference on Systems, Man and Cybernetics*, 2006.