NELSON MANDELA
UNIVERSITY

Optimisation of Humanoid Robot Walking Gaits

Philip John Winfield - 214011186

Submitted in partial fulfilment of the requirements for the degree of
Bachelor of Science Honours in Computer Science and Information Systems
in the Faculty of Science
at the Nelson Mandela University

Supervisor: Dr M.C Du Plessis

October 2019

Declaration

I, Philip Winfield, hereby declare that the treatise for the degree Bachelor of Science Honours in Computer Science and Information Systems is my own work and that it has not previously been submitted for assessment of completion of any postgraduate qualification to another university or for another qualification.

Philip Winfield - 214011186

Abstract

The project aimed to optimise walking gait on the Robotis Bioloid Premium by use of Evolutionary Robotics. An existing gait was found and improved based on the criteria of distance covered and stability. Previous success in using Evolutionary Robotics focused on using Evolutionary Algorithms to optimise gait. This meant an appropriate controller was designed to represent gait that resulted in the form of a two-dimensional matrix of motor degrees. A simulated model of the Bioloid was used to assist the evolution of the controller. The training process required a network of computers that evaluated candidate controllers before the final controller was evolved. The operators of the evolutionary algorithm were continuously refined by results achieved during the various tests and experiments. The gait on the final controller saw significant improvements to the original, both with respect to distance covered and stability. The measure of stability used a novel concept of accumulative backward sway that calculated the total swaying motion of the Bioloid throughout a gait cycle. Even though the controller proved successful in optimising the original gait in simulation, it failed to achieve locomotion when transferred to the physical Bioloid. The reality gap problem caused differences between simulation and the physical Bioloid. Solving this problem is left for future development. Ultimately, the goal was to design an Evolutionary Algorithm to optimise walking gait on a humanoid robot, which proved successful in simulation.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Evolutionary Algorithms (EA) build on the Darwinian principles of evolution, namely natural selection, common descent, gradualism and population speciation, commonly known as "*survival of the fittest*". These are then applied using stochastic optimisation algorithms [28]. The application of focus for this report is how EAs are used to develop controllers in the field of Evolutionary Robotics (ER), specifically to the optimisation of a humanoid robot walking gait. This is of particular interest to the Department of Computing Sciences at Nelson Mandela University as the department is exploring further the field of Evolutionary Robotics. Previous research has included evolving a controller using an artificial neural network as a simulator for a snake robot [30] and applying ER to develop locomotion in a hexapod robot [20]. The process of developing these controllers goes through many iterations, or generations, before the final controller is created. Through exploring the search space EAs may discover solutions, albeit non-intuitive, not previously thought of by the researcher [1, 31]. Based on the success of previous projects, applying ER to develop a humanoid robot controller for bipedal locomotion is a task that has prospective opportunities.

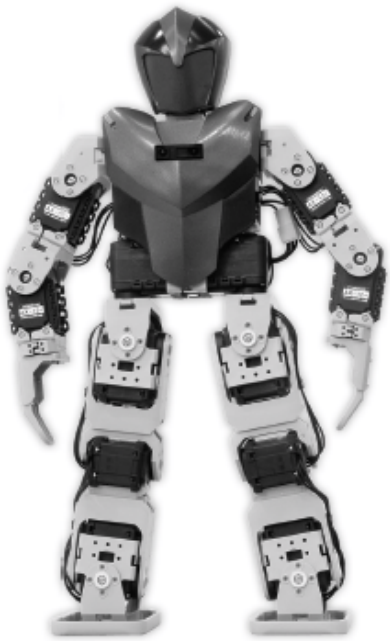
1.1 Project Overview

A humanoid robot is a mechanical device that attempts to mimic and model human anatomy in the way it functions. The morphology of humanoid robots vary when compared to each other, resulting in different models and equations to achieve bipedal locomotion. Bipedal locomotion is the ability to

move forward in an upright position with only two points of contact with the ground. The ability for a humanoid robot to move is caused by actuated hips, knees, and ankles (or any combination of human-like features). The walking gait (style) of a bipedal robot requires models and equations in order to achieve forward momentum. These models will also adapt with the changing speed and style of movement performed. [19].

The aim was to find model parameters that will improve the gait for forms of bipedal locomotion on the Robotis Bioloid Premium (Figure 1.1) based on various optimisation criteria. Robotis Bioloid Premium will now simply be referred to as Bioloid. The gait of each step or stride of the robot can be characterised by oscillations of the actuated leg and body movement [9]. Additionally, the difference in oscillation between running and walking is noticeable, hence, the parameters in a model for varying types of pedal locomotion will differ.

Figure 1.1: Robotis Bioloid Premium [25]



Ultimately, the goal of this project was to determine and optimise gait parameters for a bipedal robot, specifically the Bioloid, resulting in improved locomotion.

1.2 Project Goals

In order for the project to achieve the intended outcome, certain vital feasibility requirements and goals had to be met. The simulation would need to be set up to model the morphology of the Bioloid where the *reality gap*, Chapter 2.2.2, must be taken into consideration to accurately represent the physical Bioloid in the real-world. A gait is needed to be set up, in the form of an ER controller, and the parameters optimised by the use of an EA to improve locomotion. The evolved controller should be implemented and tested in a real-world environment where it is vital that communication, via code, is possible to the Bioloid. The following list summarised the rudimentary outline of the requirements:

- Design a controller to model gait
- Set up simulation for evolving controller
- Create an evolutionary algorithm
- Use ER process to evolve and improve the controller
- Transfer trained controller to physical Bioloid
- Minimise reality gap
- Document findings

Once sufficient results are achieved, an analysis and comparison of the controllers would need to be documented. These findings would highlight the effects the EA had when attempting to improve the gait. Relevant conclusions would be constructed based on findings and analysis of the evolved controllers.

1.3 Methodology

An *experimental* approach was used during the development of the system in order to achieve the desired outcomes for the problem goals [5]. The process was broken into two main phases. First phase consists of set up, design and evaluation; and the second phase discusses and reports results and findings [5]. Before the experimental process began an exploration into previous

attempts at robotic bipedal locomotion was researched in order to identify key aspects and questions that could to be considered. Topics relevant to the evaluation and composition of ideas, specifically those relating to Evolutionary Algorithms (EA) and Robotics (ER) were also included. There is an iterative approach in the first phase that starts by developing a framework for the controller architecture, including operations like the fitness function and other EA parameters. Then, evolving the controllers to see if they achieve the desired outcome in simulation, if not, going back to develop new ideas for the framework. Once satisfied with the results in simulation, the controllers are transferred to the physical robot and evaluated. If necessary, the whole procedure can be repeated and the framework developed further in simulation. Further research may be required to aid the iterative process of developing ideas for the framework. The final phase analyses and documents the results of the evolved controllers. Presentation of the results was important as to accurately present the findings and is a key part of the *experimental* process [5]. The discussion includes anything learnt and interesting observation that will benefit future experiments.

Chapter 2

Literature Review

This chapter forms research into the specific fields of study by exploring literature that is relevant to the task of evolving a robotic controller. Section 2.1 will provide a brief understanding of EAs in order to demonstrate how they can be used in ER and the advantages they have over conventional methods. Section 2.2 will look at ER and the two main ways in which controllers are trained, through simulation or tests on a physical robot, each with its own benefits and concerns to consider. Section 2.3 starts by explaining the concept of a gait and then looks at different ways and techniques researchers in the past have made bipeds walk. Firstly, by relating to the conventional ways of manually creating the robot's controller and walking gait, and then by exploring recent research in the use of ER. A literature review provides understanding of terminology and concepts required for the development and implementation of the research project.

2.1 Evolutionary Algorithms

Evolutionary Algorithms is a term used to describe a broad class of algorithms that are inspired by biological evolution using processes such as *selection, reproduction, mutation* and *recombination* [6]. The biological metaphor of evolution describes potential solutions as individuals; characteristics/parameters of a solution as a genome (set of chromosomes); a set of varying solutions as a population; and the creation of new solution sets as generations [28]. The evolutionary mechanisms are applied in varying ways to the metaphor which compose different EAs, such as Genetic Algorithms (GA), Genetic

Programming (GP) and Evolutionary Programming (EP).

2.1.1 Operators

The mechanisms inspired by biological evolution form the main operators that are used in the different types of EAs [6], where variations and combinations of these operators form different strategies to approach optimisation problems. EAs use an iterative process to evolve a population of individuals. The first operator that needs to be considered is which individuals are selected for *Reproduction* and make it to the next generation. This operator, *Selection*, focuses on creating continually better solutions where stronger individuals - better at solving the problem - have a higher chance of survival compared to the weaker ones [6]. Various techniques can be used during the *Selection* process in order to decide which individuals are chosen for *Reproduction*, where the focus is on what degree of fitness is considered when selecting the fittest individuals, Section 2.1.2), [6].

The process of performing *Recombination* and/or *Mutation* is referred to as the *Reproduction* of selected individual(s). During *Recombination*, otherwise known as *Crossover*, individuals, referred to as parents, perform a process that forms a child/offspring, new individual, by using components from the parents [28]. The aim is that the parameters, chromosomes, of *superior* individuals are combined together so that the offspring have the best traits of its parents [6]. *Mutation*, however, gives individuals a chance to develop better characteristic by randomly (usually with a low probability) perturbing an individual's parameters [28, 7].

2.1.2 Search Space & Fitness

The fitness of a candidate solution is a way of quantifying how well it solves the problem at hand [14]. Individuals with a higher fitness value typically have a greater chance of *selection* and *reproduction*, this follows on from the principle of *survival of the fittest* as inspired by Darwin [28]. Fitness functions are meant to help guide the population towards optimal solutions.

The search space represents the entire solution set that is possible, where initially a broad and diverse set of candidate solutions is desired in order to explore more potential solutions [6]. Initially, a population will consist of random individuals across the search space and after each generation, the population aims to converge towards a specific solution.

2.1.3 Implementation

The basic outline for a Generic Evolutionary Algorithm can be described as follows: a population of individuals is randomly generated which are then evaluated based on a fitness function. *Selection* is performed to decide which parents are chosen for *reproduction*. Offspring are created by the *recombination* of parents and then there is a chance that *mutation* is applied to them. The parents and offspring form the next generation by replacing the current population. The process of evaluation and generating a new population is repeated until a threshold for the fitness is met or a certain number of generations have been reached.

This explanation paraphrases pseudo code, Algorithm 1, to show how the different mechanisms/operators are incorporated.

Algorithm 1: Generic Evolutionary Algorithm [6]

Let $t = 0$ be the generation counter;
Create and *initialise* an n_x-dimensional population, $C(0)$, to consist of n_s individuals;
while *stopping condition(s) not true* **do**
    Evaluate the fitness, $f(x_i(t))$, of each individual, $x_i(t)$;
    Perform reproduction to create offspring;
    Select the new population, $C(t + 1)$;
    Advance to the new generation, i.e. $t = t + 1$;

2.2 Evolutionary Robotics

The previously mentioned concepts and algorithms, Section 2.1, have proven useful and powerful at finding solutions to many applications, specifically optimisation problems [28]. The area of focus is using such concepts in the field of robotics, namely evolutionary robotics (ER), to develop or optimise controllers/control policies [1]. A motivation for the use of ER training techniques is that it may produce better solutions compared to manual design.

2.2.1 Process

The ER Process uses the principles of EAs and applies them to the development of controllers and thus the terminology and process used will be synonymous to that described in Section 2.1. An individual represents a candidate solution, in terms of ER, this means an intermediate controller, where the genome representing an individual may be the parameters that effect the robot's gait or the configuration of the angle and range of motion for the robot's servos (joints) [32, 2].

The evolutionary process to train the controller would start by initialising a population set of random candidate solutions, that represent a diverse area of the search space. This newly created set becomes the current population in the iterative evolution cycle. Individuals in the current population are evaluated using the fitness function to compare their performance during a specific task. Depending on the selection criteria, *fit* parents are selected for reproduction. Offspring are created by the recombination of the parents chromosomes and mutated in the hope that offspring will contain the best characteristics of their parent(s). The offspring and their parents form a new population of candidate solutions which replaces the current population. Figure 2.1 shows the iterative evolutionary process which continues until a certain number of generations or the threshold of fitness is achieved.

Training techniques to evolve controllers are either applied in simulation or on the physical robot as commands are performed in the task environment [26]. There needs to be a way to evaluate candidate controllers whilst training which is achieved by the assignment of a fitness value. The training process is most commonly performed in simulation and only at the end transferred on to the physical robot, due to a number of reasons [15]:

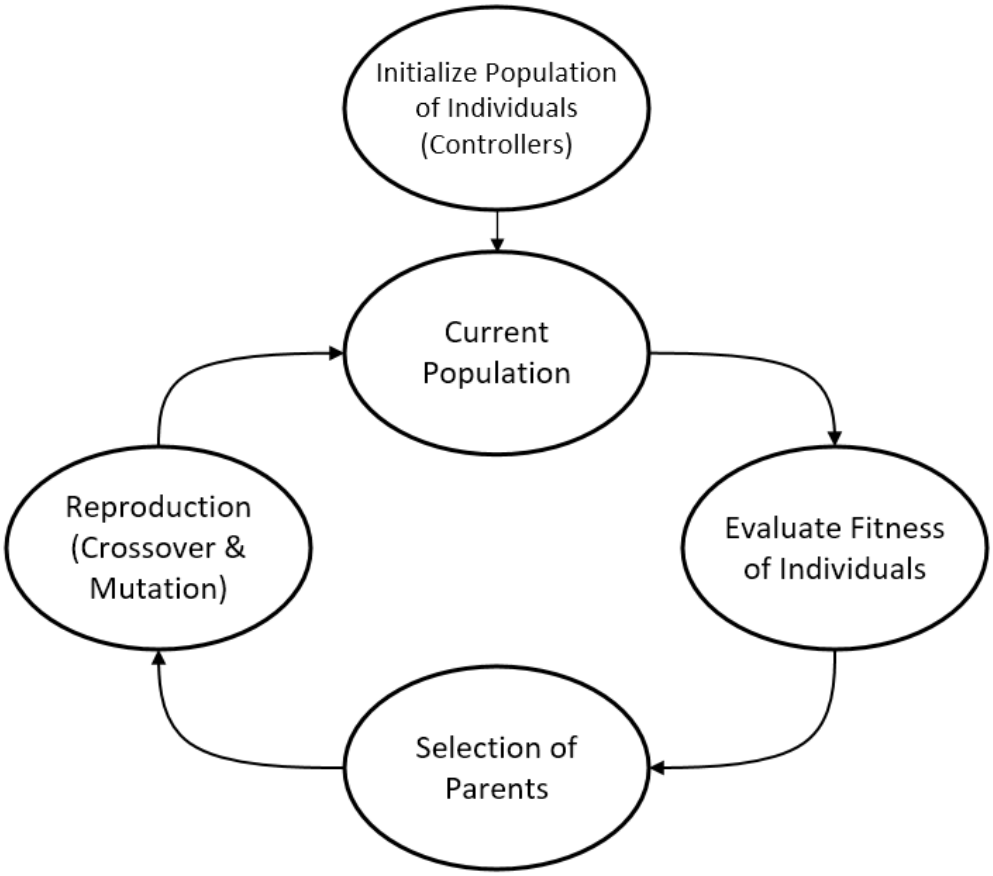
- **Mechanical Robustness**

  The vast number of individuals to evaluate means that the robot will endure continual strain on the joints and servos, specially as some individuals will have particularly damaging behaviour.

- **Energy Supply**

  The training process is time-consuming and much longer than the average battery life, therefore would need human intervention to replace batteries and charge them.

Figure 2.1: Evolutionary Robotics Process



- **Analysis** When evaluating a controller through observation, there may be researcher bias, where simulation can provide more reliable and useful feedback.

- **Time Efficiency**

  The evolutionary process of evaluating the fitness of an entire population, performing selection and reproduction for multiple generations is time-consuming, especially if done physically on a singular robot.

2.2.2 Simulation

The use of EAs require the development/training of candidate solutions, referred to as a genomes, which in the context of this project would be the parameters that effect the configuration of the robot's gait.

Evaluating controllers during online simulation is not without problems. The environment in which controllers are evolved typically use physics simulators where the inaccuracies or simplicities of their design may be exploited whilst training. This leads to differences in the physical robot from that of the evolved controller. Complications from training to what is presented in the real environment is known as the *reality gap* problem [12]. Fortunately there are solutions to overcome the *reality gap* and one such way is the injection of *noise* in simulation [1]. By slightly altering values of the robot's sensors, motors and position, referred to as *noise*, helps prevent the exploitation in training as previously mentioned.

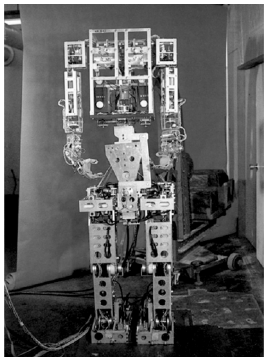
2.3 Bipedal Robotic Locomotion

Unlike most wheeled robots, legged robots do not rely on a surface that is continuous and flat therefore they have increased mobility on uneven terrain [7]. Bipedal locomotion is a fundamental task for a humanoid robot and is therefore important in its design. The first full-scale humanoid robot to achieve bipedal locomotion was the WABOT-1 (Figure 2.2a) in 1973 designed at the Waseda University [11]. Another significant project before the turn of the century was P2 (Figure 2.2b) designed by Honda in 1998 [8]. P2 had the ability to walk up/down stairs and was the first cable-less humanoid robot. Honda then went on to create the iconic ASIMO in 2000, which had the ability to change directions whilst in continuous motion [13]. More recent projects include HUBO (Figure 2.2c) by KAIST and Atlas (Figure 2.2d) by Boston Dynamics which started in 2013 with progressive developments in the ability of the robot such as: running, walking over uneven terrain, jumping over obstacles, and even performing backflips [17, 4].

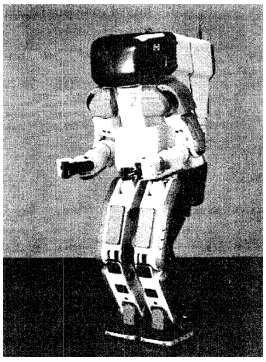
Two main strategies have been used to generate the gait of legged robots [7]. The first is based on the assumption that humans have an optimal gait, therefore, try to mimic human gait on a robot. The second views gait generation as an optimisation problem that is multi-constrained and multi-

objective, and therefore, tries to develop gait by optimising performance criteria.

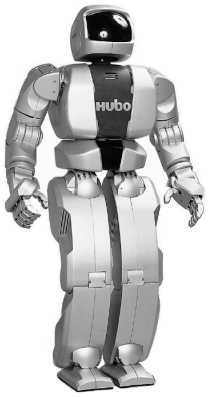
Figure 2.2: Humanoid Robots



(a) WABOT-1 [11]



(b) P2 [8]



(c) HUBO [17]

(d) Atlas [4]

2.3.1 Brief Understanding of Gait Design

The gait cycle, otherwise known as a *stride*, is broken into two periods, *stance* and *swing*, consisting of a total of eight phases [18]. The combination of phases from Figure 2.3 and Figure 2.4 make up the gait cycle. Furthermore, certain tasks are achieved throughout the phases to accomplish locomotion and account for a percentage of the cycle. The task percentage can change within a certain range depending on how the gait is set up. Figure 2.5 shows

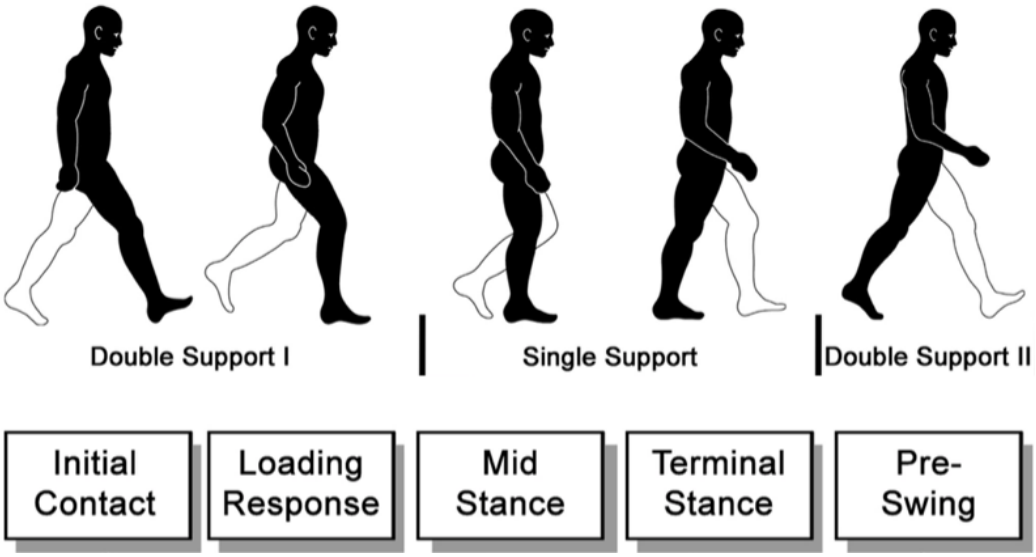
how the gait cycle is divided into each component.

Stance Period

As seen in Figure 2.3, the stance period accounts for all phases the foot has load-bearing contact with the ground. In the stance period, there are single and double support phases where either there is one leg in contact with the ground or two, respectively, which provide stability and support the weight of the person/robot.

- Initial Contact - moment the foot strikes the ground
- Loading Response - the limb begins bearing weight
- Mid-stance - the limb is the only support of the body
- Terminal-stance - last phase of being only support
- Pre-swing - support shared and limb prepares to swing

Figure 2.3: Stance Phases [27]

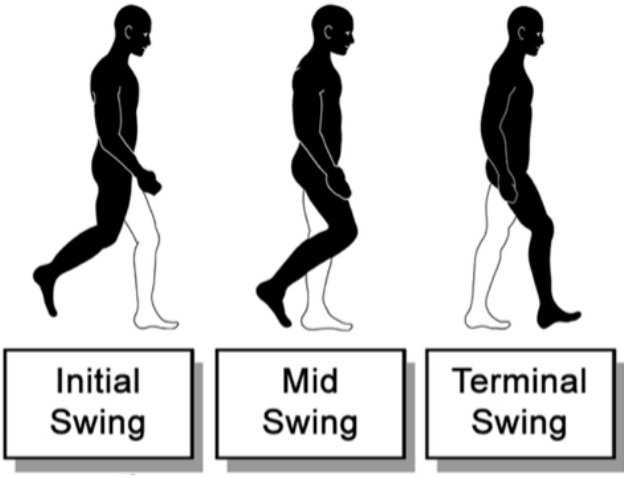


Swing Period

As seen in Figure 2.4, the swing period accounts for all phases the foot is in the air without any ground contact. The trajectory of the foot is determined by the initial and mid-swing, then termination swing prepares the foot for the stance period.

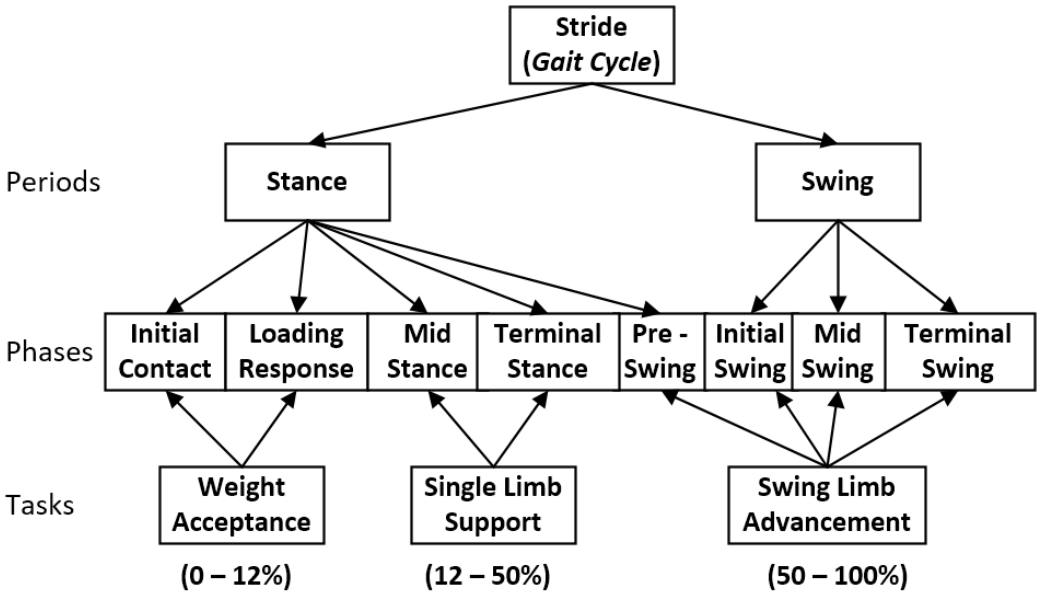
- Initial swing - limb begins to swing and advance
- Mid-swing - thigh reaches peak advancement
- Terminal-swing - limb prepares for ground contact

Figure 2.4: Swing Phases [27]



How each of the two periods are divided into the gait cycle is represented in Figure 2.5. As seen the stance period is responsible for holding the weight, either by beginning to bare load on a limb or using a single limb to bare the entire load, and the swing phase has the task of gaining distance with the swing of a limb.

Figure 2.5: Divisions of the Gait Cycle [18]



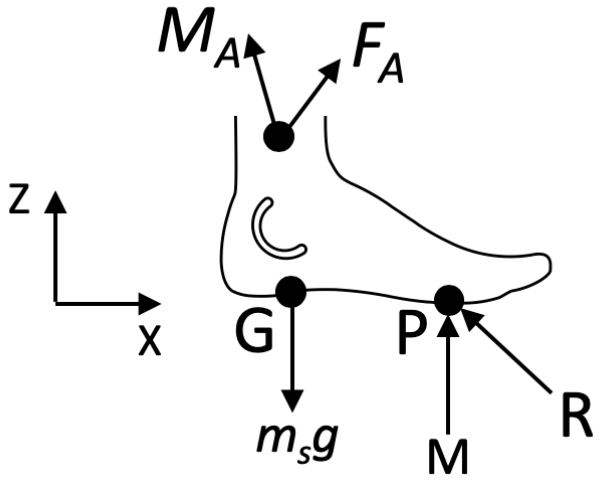
The terminology of the gait design in humanoid robots is synonymous to that presented here of humans. The morphology of a humanoid robot does not necessarily represent the joints and limbs of a human accurately, therefore, the degrees of freedom (DOF), independent directions of motion, of a robot may alter the gait design.

2.3.2 Zero Moment Point

An important task in the gait design of bipedal locomotion is that the humanoid robot should maintain dynamic balance, in other words, stability [29]. As seen in Section 2.3.1, the single and double support phases mean the robot is either supported on one or both feet during the different cycles of the gait. All mechanisms of joints in a humanoid robot are programmable and can be controlled except for the contact the foot has on the ground. This lead to the development of Zero Moment Point (ZMP) by Vukobratović and Juričić, which is a point where all influencing forces can be replaced by a single force [29]. The ZMP is a point on the ground where the net moment of inertial forces and gravitational forces have no horizontal component, Figure 2.6, indicates this point (P) and the forces acting upon the foot. Where F_A

and M_A represent the collective force and moment of components above the ankle, point G is the weight of the foot at its gravitational center, R and M at point P represent the force and moment of the total ground reaction. For simplicity, Figure 2.6, only shows the z and x axes, where in fact all the forces and moments are composed of x, y and z axis components.

Figure 2.6: Zero Moment Point [29]



Provided that the foot does not slide, friction represents the horizontal components of R, (R_x and R_y) and balances out the horizontal components of F_A, while the R_Z force balances out all the vertical forces acting upon the foot. The moment of friction is represented by M, where M_z, balances out the vertical moment of M_A and that induced by F_A. The necessary conditions for the horizontal components of M to achieve dynamic equilibrium should be:

$$M_x = 0, \tag{2.1}$$

$$M_y = 0. \tag{2.2}$$

Where the position of the ZMP to ensure dynamic equilibrium, whilst the foot is fully flat on the floor, should satisfy the following equation with all the forces and moments acting upon the foot:

$$R + F_A + m_s g = 0 \tag{2.3}$$

It should be noted that as the robot's Center of Gravity changes and all the forces are shifted the position of ZMP also changes [29]. The ZMP has been an integral part in the development of humanoid robots and has been evidently included in the design of ASIMO, HUBO and RoboSapien to achieve dynamic locomotion [8, 17, 32].

2.3.3 Use of ER

The complicated morphology of a legged robot means there are multiple DOFs and variables to be considered. This results in many parameters that need to be established, for instance, the Robotis Bioloid Premium (Figure 1.1) has 18 servos where each represents a DOF [25]. This supports the strategy of viewing gait generation as an optimisation problem, that is multi-constrained and multi-objective [7]. The number of possible solutions and variations mean that the parameters form a multi-dimensional solution space, where standard optimisation methods can become very complex or may take longer to solve. Therefore, the use of ER is considered to help the process of optimising the gait for bipedal locomotion. Such as, using EA to fine tune the parameters of a predetermined trajectory in a gait [32].

In order to optimise the gait using ER, the evaluation of fitness on the gait must be considered. The criteria for fitness depend on the objective function and is an important part of EA, Section 2.1.2. In terms of gait optimisation there may be various objectives that are considered which present multi-objective functions [7]. For instance, if the gait is focusing on optimising the *maximum velocity*, the fitness function can be set up to represent the displacement of the robot after a designated time, see Equation 2.4 [22].

$$F = \sqrt{(x_t - x_0)^2 + (y_t - y_0)^2} \tag{2.4}$$

Where F uses the Pythagoras Theorem to calculate the distance between the initial (x_0, y_0) and final (x_t, y_t) points of the robot.*Stability* is another important objective as it ensures the robot will be able to move without falling over. Dynamic stability can be verified by the ZMP, Section 2.3.2.

Fitness functions incorporate the ZMP in order to focus on developing stable gaits, see Equation 2.5 [32].

$$F = \frac{1}{\sum_{n}^{i=1}(x_{zmp}(i) - x_{dzmp}(i))^2} \tag{2.5}$$

Where x_{zmp} is the ZMP derived from ankle torque and the x_{dzmp} is the distance between the ZMP and ball of the foot.

The gait is a cyclical repetitive function, Section 2.3.1. In order to ensure *smooth transitions* from one cycle to the next, the start and end positions, as well as velocity, need to be the same [7]. This criterion can be added to the fitness function to ensure a natural repetitive gait. Another objective is to minimise *consumed energy* that is used during the gait cycle. Not only does it produce a more natural gait, as *consumed energy* is minimised in a human gait, it also maximises the battery operation time [18, 7].

2.4 Conclusion

The principles and techniques used in Evolutionary Algorithms follow the metaphor of natural evolution and forms the basis of Evolutionary Robotics. The morphology of a humanoid robot mimics the human anatomy.Thus, the study of gait design can benefit the development of the EA. Through literature, it is seen that ER can be applied to the optimisation of bipedal locomotion, where the criteria is to improve distance and stability.

Chapter 3

System Design

This chapter follows on from the knowledge gained in Chapter 2 and applies it to the development of a system to evolve a controller to achieve optimal bipedal locomotion. The chapter focuses on the different components of the system that are required to work together. Firstly, the simulation used during training is explained in Section 3.1. Then follows onto the system architect of improving the training process by maximising the computational power in Section 3.2. Section 3.3 describes how the trained simulated controllers were transferred onto the physical robot. Section 3.4 focuses on the different representations that were considered to define the gait on the controller. Each component is carefully thought through and the available resources utilised.

3.1 Simulation

Evolving the controllers in simulation is a vital part to training since it reduces time taken, as well as providing for a safer environment to test potentially harmful controllers, to the hardware on the physical robot. The simulation software used was V-rep, Figure 3.1, developed by Coppelia Robotics [24]. The reason V-rep was used over other simulation software such as Gazebo, Webots, RobotStudio, Virtual Robotics Toolkit, etc. was due to various contributing factors. Table 3.1 shows a comparison of the main simulation software that was considered.

Figure 3.1: V-REP Simulation Software

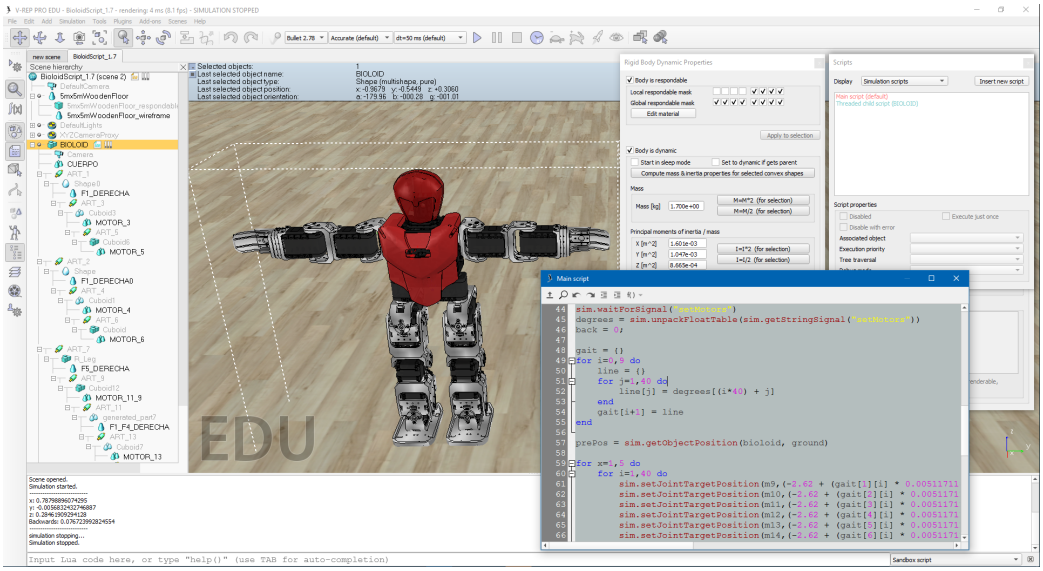


Table 3.1: Simulation Software Comparison [24] [23] [10]

	V-REP	**Gazebo**	**Webot**
Platforms	Linux, macOS, Windows	Linux, macOS, Windows	Linux, macOS, Windows
Physics Engine	Bullet, ODE, Vortex and Newton	Defaulted to ODE	ODE
External APIs	Multiple including JAVA	C++, ROS	Multiple including JAVA
Internal Programming Language	Lua	C++	C++
Existing Models & Scenes	Yes	No	No

Gazebo was unable to be installed on Windows and was, thus, limited to Linux and macOS [23]. Gazebo required programming restricted to C++ and ROS. Conversely, V-REP and Webots included extensive external APIs which included the more favourable JAVA. V-Rep allows the use of a more

diverse range of physics engines which would improve transferring the simulation to a real-world environment. A significant factor supporting the use of V-REP was the discovery of a repository by Correia which included a model of the Robotis Bioloid with the same dimensions, and Degrees of Freedom for a scene in V-rep [3]. Whilst the repository contained code for communication between Java and V-rep using the Coppelia API, the program was only used to reference the different functionality of the API and a basic gait for locomotion.

3.2 System Architecture

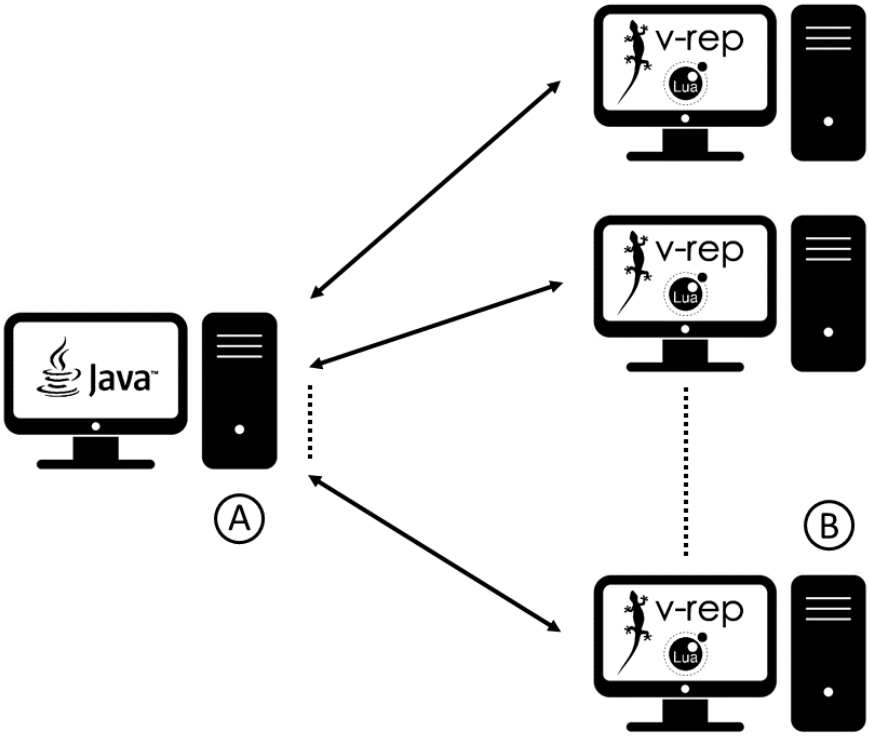
Problems faced during the development of the Evolutionary Algorithm lead to the construction of a network of computers that aided the training process. The first concern was the time the EA took to evaluate candidate controllers. During the evaluation phase, the motor degrees that represented the gait of a particular controller was sent to the model in V-rep via Coppelia's API. The Bioloid model would then perform the gait and the results captured. The problem was that V-rep would only accept communication through a single port and only one controller could be evaluated at a time. This was particularly concerning as the time taken to evaluate a controller was up to 15 seconds, if the Bioloid successfully walked throughout the entire evaluation. Therefore, using a standard population size of 100 and 1000 generations training would take about 17 days. Owing to the experimental nature of the methodology, many variations to the methods in the algorithm needed to be tested, as well as refining which parameters produced the best results. Evaluation time needed to be significantly reduced. The solution was to find a way to allow multiple instances of V-rep running at the same time and distribute the workload between them. This could be achieved by either opening multiple windows of V-rep on the same computer but changing the communication port between each. Or, by using many different computers that each ran V-rep where a computers IP address could distinguish them. The latter was chosen as multiple instances on one computer required more computational power, and a computer laboratory was available for use at the Nelson Mandela University.

A further hurdle to overcome was how the program sent commands to each instance of V-rep. Initially, the program would send the angle degree, of

each motor, for every timestamp in the gait cycle. This proved problematic as threads were being used to simultaneously and continuously send commands to more instances of V-rep, which overloaded the computers' CPU. Apart from the hardware issue, this also led to inconsistencies in the evaluation of the controllers. Where, the minor time delays between which thread sent a command, meant that the gait in the simulation was not smooth. The solution was to write a Lua Script for V-rep, that received the entire command sequence of a controller. Then, after evaluating the controller would send back the result. This meant that each thread dealt with significantly less communication, which settled the CPU, as well as producing consistent results.

The final architecture during the evolutionary process can be visualised in Figure 3.2. A single computer, A, dealt with the entire evolutionary computational processes of *initialisation, selections, mutations,* and *crossovers* of each candidate controllers. Computer A would also create a connection to the various instances of V-rep, where each reference was added to a pool of computers, B, that the program could utilise to evaluate candidate controllers during the evolutionary process. Each computer in the pool, B, had a scene in V-rep with a Lua script, waiting to receive a controller to evaluate. During the evaluation phase in the EA, the program would take a batch of candidate controllers. Then, using the pool of available simulations evaluate each batch of controllers until the entire generation had been evaluated. Computer A would perform the rest of the processes in the EA. When another generation of candidate controllers needed to be evaluated, the process previously described would be repeated.

Figure 3.2: Network Architecture [1]

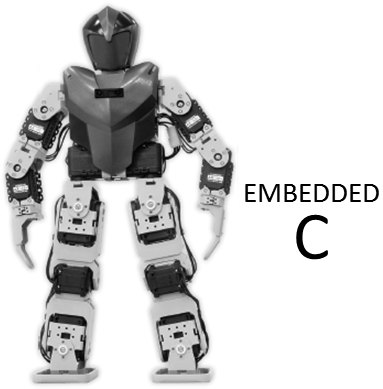


3.3 Physical Robot

During this phase of development, the initial set up of transferring the trained controllers to the physical robot was implemented. Using the Robotis e-Manual the firmware on the CM-530 (the onboard computer on the Bioloid) was overridden to transmit an Embedded C program that contained a trained controller [25]. The Embedded C program built on code found in examples from the Robotis e-Manual, where the main method was to cycle through the gait of a controller. During the development of the EA, differences between the trained simulated controller and the controller on the Bioloid required refinement to minimise differences between simulation and the real robot.

[1] Figure created by author using logos retrieved online [16] [21] [24]

Figure 3.3: Embedded C [25]



3.4 The Controller

The controller describes the gait and is what the EA of the program is ultimately aiming to optimise. During the experimental phase three different types of controllers were considered. The first represented the gait as a series of polynomials for each of the motors on the Bioloid used in locomotion. The second attempted to use trigonometric functions in the form of sine waves to represent the gait. Both these controllers would calculate the motor degree of each motor during the gait cycle. Which led to the last controller simply representing the gait as a series of motor degrees for each motor.

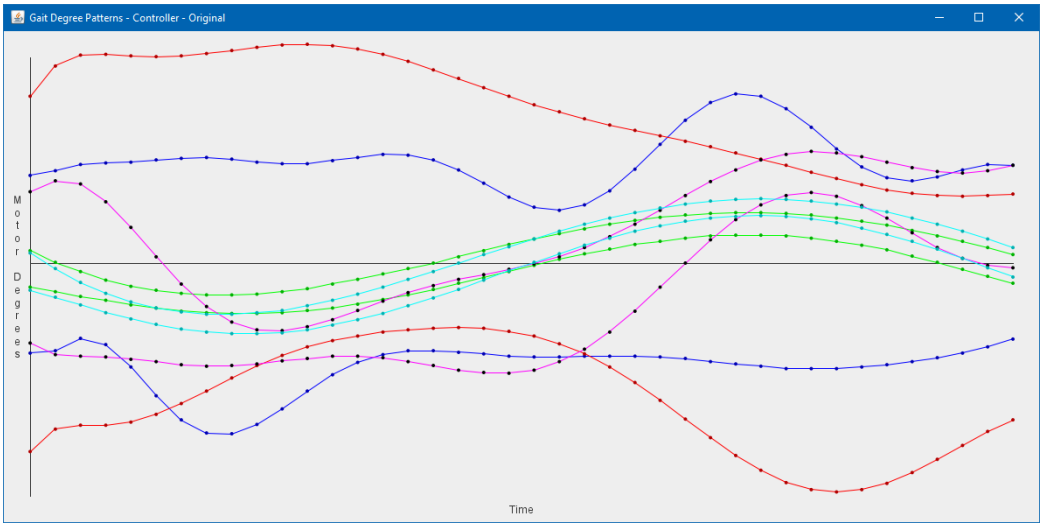
3.4.1 Polynomial

The original gait found in the repository represented the gait as polynomials up to the 14th order, and thus became the starting point of attempting to optimise the coefficients. Each motor had a unique equation where Equation 3.1 shows the series of polynomials. The variable $f_i(x)$, is the calculated motor degree for the corresponding x value, which is the timestamp of the position in the gait cycle.

$$
\begin{aligned}
f_1(x) &= c_{1,14}x^{14} + c_{1,13}x^{13} + c_{1,12}x^{12} + ... + c_{1,2}x^2 + c_{1,1}x + c_{1,0} \\
f_2(x) &= c_{2,14}x^{14} + c_{2,13}x^{13} + c_{2,12}x^{12} + ... + c_{2,2}x^2 + c_{2,1}x + c_{2,0} \\
&\vdots \\
f_{10}(x) &= c_{10,14}x^{14} + c_{10,13}x^{13} + c_{10,12}x^{12} + ... + c_{10,2}x^2 + c_{10,1}x + c_{10,0}
\end{aligned} \tag{3.1}
$$

The more variables added to the polynomial the more accurately the shape of the gait can be fitted. This meant the equations fitted a very smooth and undulating gait cycle, as seen in Figure 3.4. The colours match opposing motors of each leg, for instance, the red lines represent the respective left and right knees.

Figure 3.4: Original Gait



Even though polynomials could represent the gait cycle the problem came when trying to optimise the coefficients. Owing to the nature of high order polynomials the coefficient had a wide range of values, $0.0000001 < c < 20000$, and during mutation/ crossovers the slightest change to the higher-order x values would drastically change the shape of the graph. Thus, either a more complicated method with restrictions to the mutation/ crossovers was needed, or a simpler equation to represent the gait.

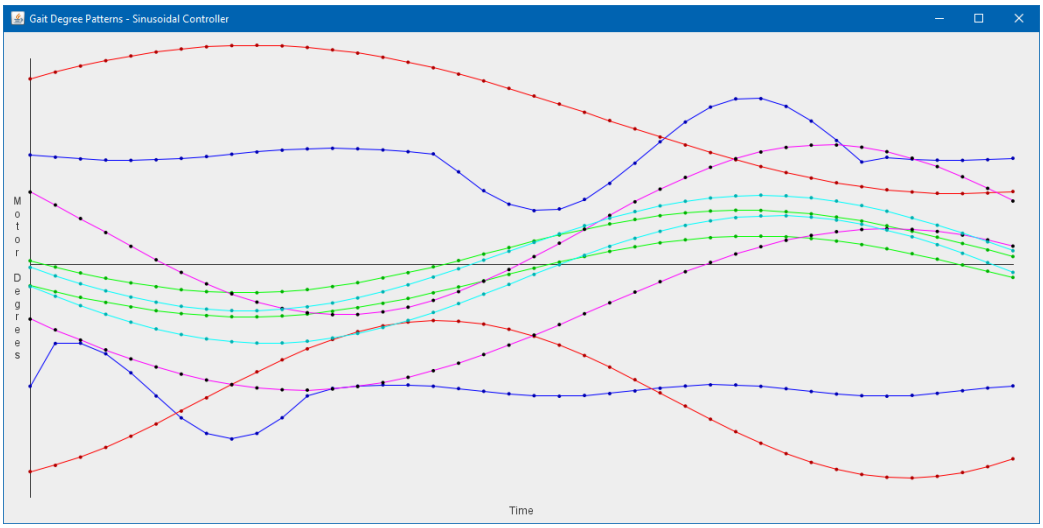
3.4.2 Trigonometric

Through observations of the curve traced by the graph seen previously, and the repetitive nature of the gait, the idea to use sinusoidal graphs was considered. Using the form of Equation 3.2, the gait was initialised for each motor where constraints to the parameters were added.

$$f(x) = a + sin(b^x + c) + d \tag{3.2}$$

Attempts to match the graph of the original gait using a sinusoidal function proved successful for some of the motors but, unfortunately unsuccessful for others. Figure 3.5 shows the result of this attempt, though it comes close, it does not accurately represent the gait.

Figure 3.5: Sinusoidal Controller



Literature has shown that the gait is commonly described by a Fourier series, the summation of multiple trigonometric equations, therefore the use of a single sinusoidal equation would not suffice [7].

3.4.3 Matrix

During the construction of the controllers, represented by polynomials and trigonometric equations, it was still required to calculate the motor degrees for each motor. Thus, instead of using equations, a controller was set up that

represented the motor degrees as a matrix. The matrix in Equation 3.3 was used, where $m_{i,j}$ represents the degree of motor i and timestamp j.

$$\begin{bmatrix} m_{1,1} & m_{1,2} & \cdots & m_{1,j} \\ m_{2,1} & m_{2,2} & \cdots & m_{2,j} \\ \vdots & \vdots & \ddots & \vdots \\ m_{i,1} & m_{i,2} & \cdots & m_{i,j} \end{bmatrix} \tag{3.3}$$

Any changes to the motor degrees would shift and morph the graph into different shapes where the range of degrees would be fairly similar making mutations and crossovers more stable.

3.5 Conclusion

The training of the EA required various components to work together. Firstly, simulation aided in the training of the EA. The software used was V-REP where it made use of a model representing the Bioloid. The network design of computers that ran the simulation software, was necessary to increase the computational power required during training. The design of the controller was important as the EA aimed to optimise parameters and thus various representation was analysed. The comparisons led to a controller that used a matrix of motor degrees as the representation of the controller. The simulated controller needed to be implemented on a physical robot which meant the original program on the Bioloid controller was overwritten.

Chapter 4

Algorithm Implementation

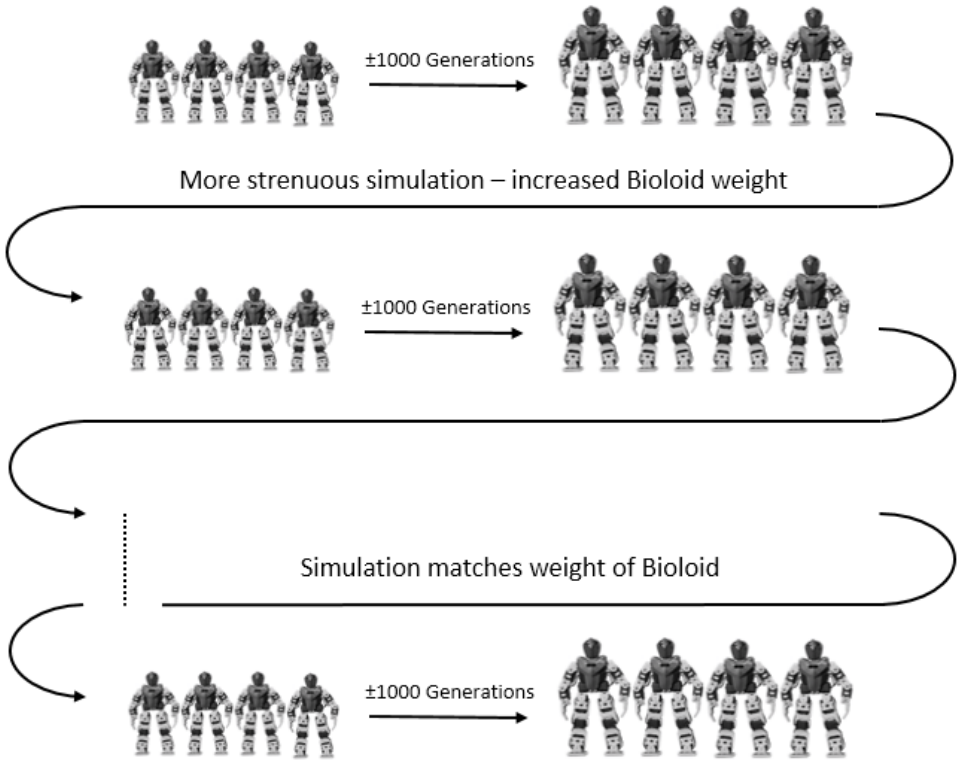
Chapter 4 explains the Evolutionary Algorithm that was used to train the controllers. The structure of the EA followed principles of a Genetic Algorithm where the pseudo-code and operations were explored in Chapter 2.1. Due to the nature of continuously improving the design of the algorithm, each section explains the preliminary designs and how they were modified and improved until the final version. Section 4.1 begins the discussion on the various operations by explaining how the population of controllers was initialised. The comparison between controllers required the appropriate design of the fitness function which is explored in Section 4.2. The selection process is discussed in Section 4.3 as individuals were chosen for reproduction. Sections 4.4 and 4.5 explore the effects of crossovers and mutations and how they influence the reproduction of offspring. While the chapter explains the modifications to the operators, the results that support the changes come from the analysis of results in Chapter 5.

4.1 Initialisation

The first step in GAs is to initialise the first generation. The aim of the project was to optimise an existing gait and to improve the gait based on some criteria. Therefore, initialisation was not completely random. Instead, mutations were performed on an existing to help guide the exploration of the GA. The method of mutations is described in Section 4.5 to follow. After several tests using the original gait to form the initial population, it was discovered that the model representing the Bioloid in simulation was lighter

than the physical robot. Thus, the weight needed to be increased in simulation. This caused problems to the initialisation, since the original gait would always cause the model with a heavier weight to fall over and the EA would get 'trapped' in a bad solution. To overcome this, the model in the simulation would incrementally increase the weight. Train a new generation which would become the initial population for a new scene with a slightly heavier model. The process continued until the weight of the simulation matched the physical Bioloid. Figure 4.1 shows this process where the first initial population used variations of the original gait as previously described.

Figure 4.1: Initialisation

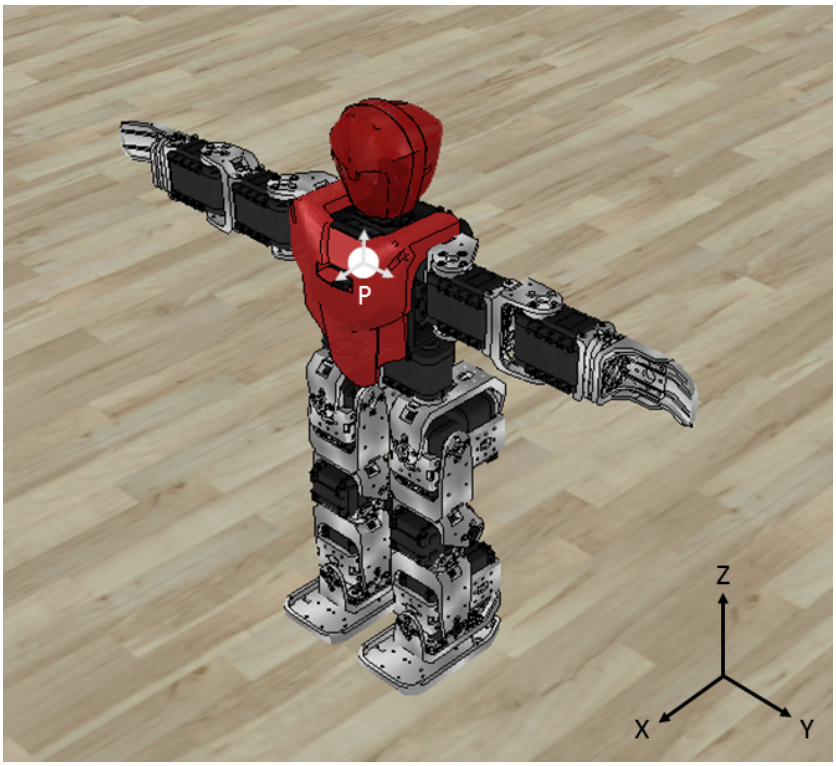


4.2 Fitness Evaluation

During the literature review various methods to evaluate how well a controller performed were studied. The first considered the ZMP, Section 2.3.2, that

could be used to create an optimal gait in terms of the stability. However, this required a more detailed model that incorporated forces acting on the Bioloid's foot. Unfortunately, the model did not contain that level of detail and, therefore, other fitness criteria needed to be considered. The second, as mention in Section 2.3.3, used the displacement of a robot using the Euclidean distance. The distance the Bioloid covered during evaluation become the criteria for comparing the candidate controllers, based on the displacement of point P in Figure 4.2. Point P consists of the x, y and z components of the position of the Bioloid.

Figure 4.2: Bioloid Orientation - Screenshot of model in V-rep



The original gait had the Bioloid yawing to the right side and ended up walking in a large circle. The first goal was to improve the gait by awarding controllers that walked in a straighter path. Therefore the fitness function described in Equation 4.1 was tested.

$$F = \Delta x - |\Delta y| \tag{4.1}$$

By subtracting the absolute value of y the fitness function penalised controllers that veered to the left or right. This proved successful in training. Though, unstable results where the Bioloid fell over, had an equal chance of reproducing as well exploiting the momentum of falling forward to gain more distance. The next iteration of the fitness function checked to see when the Bioloid fell over, whether or not the z component of point P went below a certain threshold, and added a penalty to the fitness function if it did, Equation 4.2.

$$F = \begin{cases} (\Delta x - |\Delta y|) - 0.35, & z < 0.25 \\ (\Delta x - |\Delta y|) & z \geq 0.25 \end{cases} \tag{4.2}$$

The value of 0.35 was calculated by the additional distance the Bioloid covered when it fell over and the constraint of 0.25 is roughly the Bioloid's tipping point. Therefore, the fitness function represents the total distance the Bioloid moved before it fell.

The final addition to the fitness function was to improve the stability of the robot when walking. The focus was to penalise the robot for swaying forwards and backwards whilst walking. During evaluation, the current position was compared to the previous position. If the current was less than the previous position, the robot's torso was moving backwards. The difference was added to a tally to calculate the total distance the robot swayed backwards, Accumulative Backwards Sway (ABS), Equation 4.3.

$$ABS \mathrel{+}= \begin{cases} (x_{i-1} - x_i), & x_i < x_{i-1} \\ 0 & otherwise \end{cases} \tag{4.3}$$

Then the ABS is subtracted from the fitness function as a penalty. Using the fitness function described in Equation 4.2 the final fitness function is seen in Equation 4.4

$$F_{new} = F - ABS \tag{4.4}$$

4.3 Selection

The process to select which individuals were used for reproduction was based on the principle of Tournament Selection. The first iteration divided the

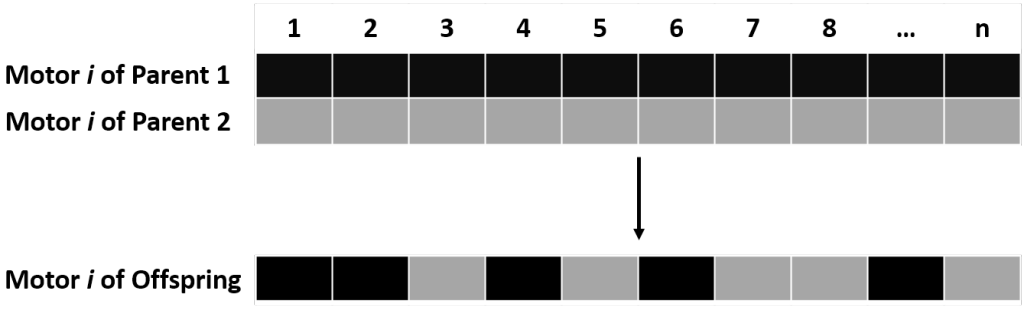
population into 10 groups and selected the best individuals and placed them immediately in the next generation. Then those 10 individuals would select pairs at random to create an offspring, until an entirely new generation had been populated. This focused on exploitation rather than exploration as a significant amount of inbreeding occurred. It also happened to be a slight misinterpretation of Tournament Selection and corrections were made.

To overcome the amount of inbreeding and encourage exploration, the selection process was refined. Similarly, the top 10 individuals were placed immediately in the next generation but the creation of offspring differed. By randomly selecting a group of 10 individuals the algorithm would select the fittest individual to become a parent, this process was repeated so that two parents were selected for reproducing an offspring. This way of selecting parents for reproduction was repeated until there was a new generation.

4.4 Crossovers

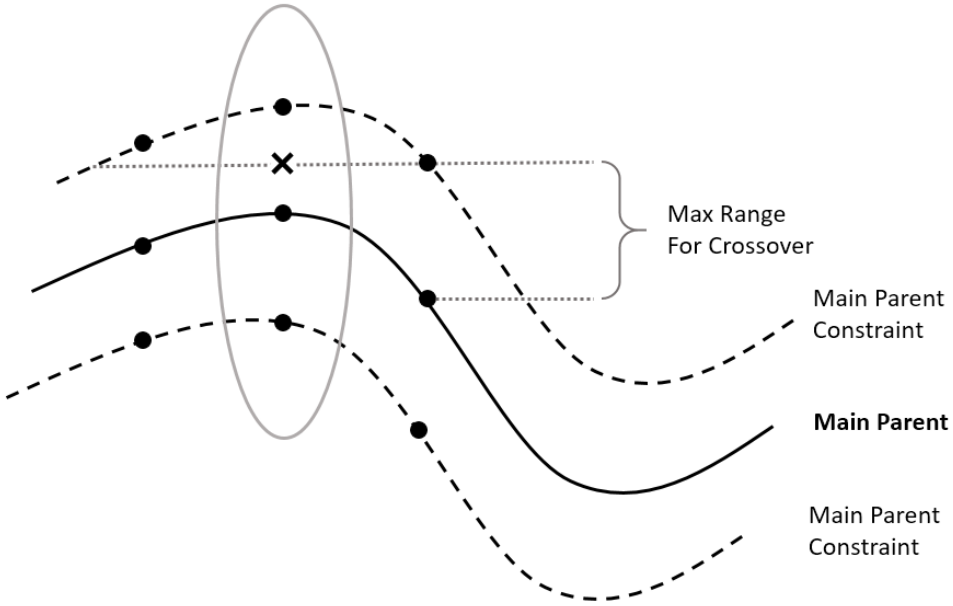
After two parents had been selected a child was reproduced by performing crossovers and mutations. An example of the controller that used motor degrees as the characteristics of the gait will be used to explain the crossover process. Each motor degree of the controller is seen as a gene which results in a two-dimensional matrix representing the chromosome. The first dimension being the different motors on the Bioloid, while the second is the change in motor degrees for each motor. For each motor in the first dimension, uniform selection would be performed to create a new array of motor degree changes, genes, for that motor. There is an equal chance for the motor degree of each parent to be selected to create the chromosome of the offspring. Figure 4.3 illustrates this by using the black and grey blocks to represent the array of genes for each parent respectively and then the offspring would be a combination of them both. Uniform crossovers would be performed for the motor degrees for each motor.

Figure 4.3: Uniform Crossover



An improvement was made to the crossover process since the combination of motor degrees of each parent would sometimes produce large jumps from one motor degree to the next. To overcome this issue, one parent was seen as the main parent during crossovers. When the second parent's gene was selected the method would compare it's value to the main parent and make sure it fell within an acceptable range of motion. Figure 4.4 illustrates the constraint as parallel lines where the motor of focus, highlighted by the oval, has a maximum range of motion calculated by analysing the position it came from and is going to. For simplicity Figure 4.4 only indicates the maximum range of values that fall above the main parent, where 'x' is seen as the upper limit. This means that if the secondary parents value is chosen for the child it must fall within the range of the main parent's value and the upper/(lower) limit otherwise it will be constrained to the value of 'x'. The addition of constraining values during crossovers significantly helped reduce the drastic changes in motor degrees.

Figure 4.4: Main Parent Crossover Limit



4.5 Mutation

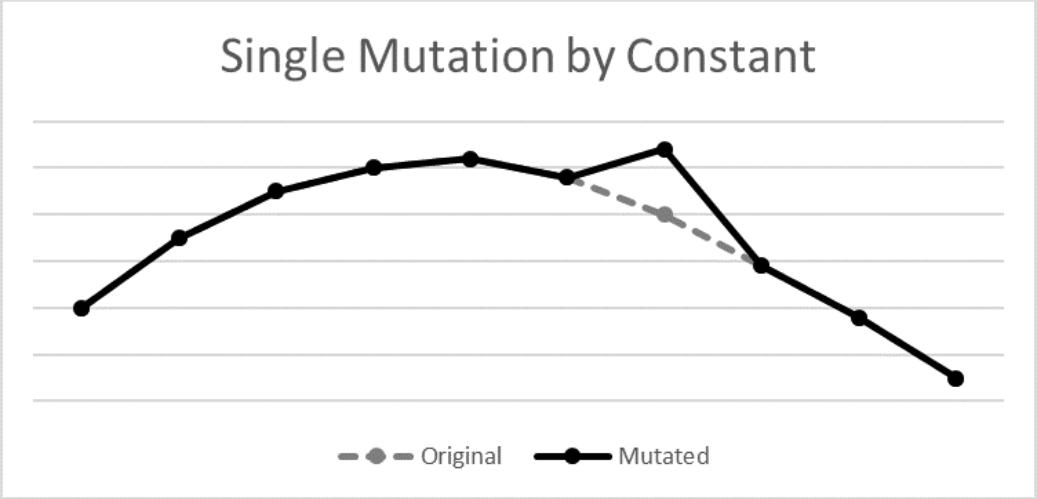
The mutations on a controller are essential to the exploration and exploitation of the solution space by introducing new characteristics to an individual [6]. Along with crossovers they are responsible for creating diverse solutions by changing a controller to produce different results in the evaluation. As they are important to the evolutionary process in optimising the resultant controller many interactions and improvements have been made. Throughout each iteration, the mutation rate of a gene was kept relatively small, between 5-10%, which meant that the probability of an individual including some mutation was about 87-99%, based on Equation 4.5 [6]. Where x_i is an individual in the population, p_m is the probability a gene is mutated and n is the number of genes in a chromosome.

$$Prob(x_i \; is \; mutated) = 1 - (1 - p_m)^n \tag{4.5}$$

The first iteration of mutation simply added a constant to the original motor degree/ gene. The constant was multiplied by a random value fol-

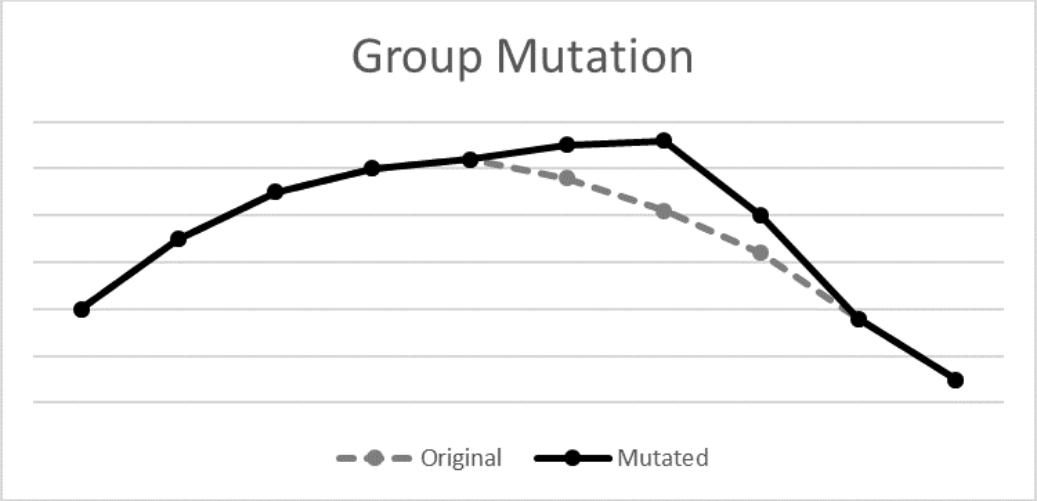
lowing a Gaussian distribution between (-1, 1) to allow for changes in both directions. The graph in Figure 4.5 indicates how the shape of the graph of motor degree changes would transform. Unfortunately, single changes would produce disjointed motor degree changes that were not smooth. The results of this mutation will be examined in more detail in Chapter 5.

Figure 4.5: Single Mutation



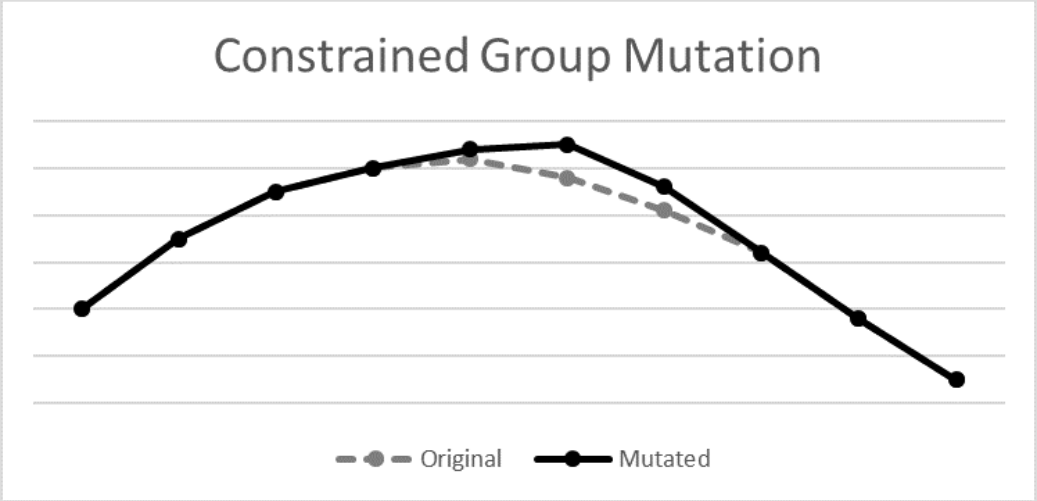
The second attempted to smooth out the changes by mutating the surrounding motor degrees by a smaller ratio of what the current gene is being mutated by. For instance, if gene x_i is being mutated by a factor of k, then gene x_{i-1} and x_{i+1} would be mutated as well by $\frac{1}{3}\ k$. The mutations that were produced are similar to that represented in Figure 4.6.

Figure 4.6: Group Mutation



This method of mutation significantly produced smoother new characteristics to the chromosome. Though as the results indicate in Chapter 5 further refinements needed to be added as the mutation method still allowed for large jumps between motor degrees. This would present a problem when translating the simulated controller to the real robot as additional strain on the motor would be presented as well as potential for slips in the motors that reduce stability. Constraints were added to the mutation method to account for any large leaps between motor degrees. To determine the constraint that the change could be, the algorithm would calculate the average change in the motor degree of the motor before and after and use a random percentage of that as the change. This was to accommodate for the differences throughout the gait cycle, for instance, if the degrees were increasing rapidly or plateauing. The graph in Figure 4.7 shows constrained values compared to that of the Figure 4.6.

Figure 4.7: Constrained Mutation



A further improvement was to make sure that the start of the gait cycle lined up with the end. Therefore mutations also accounted to reduce the differences between the initial and final positions of the motor degrees in a gait cycle by guiding mutations towards each other.

4.6 Conclusion

Each operation in the EA played an important role in its ability to achieve optimal results so it was important to refine the methods. The initialisation used an existing gait to create a starting population. Then the trained population would be used as the initial population for the next experiment. The fitness function focused on the distance covered while adding a penalty for less stable controllers. Based on the fitness of candidate controllers the selection of parents for reproduction used tournament selection. The uniform crossover process was adapted to accommodate for extreme changes in the degrees of the motor by adding constraints. Analysing the representation of the controller meant the mutation method was carefully designed to give more stable mutations. The experimental approach enabled the operations to be continuously revised and improved.

Chapter 5

Results

Previously, Chapter 4, the methods and functions in the implemented evolutionary algorithm were explained and how they were changed to improve the results. This Chapter will focus on the results during the various iterations of the algorithm that motivates these changes. Section 5.1 uses the distance travelled as the criterion to evaluate the candidate controllers. As it was later discovered, the model in simulation did not accurately represent the physical Bioloid, therefore, an additional criterion of stability was added whilst improving the simulated model. Section 5.2 explores the effects of using the stability of the simulated Bioloid in the fitness function. The trained controllers were transferred onto the physical Bioloid and the observations were analysed in Section 5.3. The results make use of data retrieved through the simulation software, mainly the coordinates throughout the performance of the gait, as well as a graphical representation of the gait on the controller. In addition, visual observations have been a key element in analysing the performance, where videos of the Bioloid in simulation and on the physical robot are referenced and can be accessed through a private YouTube link[1]. The video mentioned will now simply be referred to as the results video. The results video begins by showing the network architecture of how a computer laboratory at Nelson Mandela University was utilised for the training process and then the original gait as a point of reference. Finally, in order to see if the EA did in fact improve and optimise the original gait a comparison between the original gait and final evolved gait is discussed in Section 5.4.

[1]YouTube url: https://youtu.be/twI3oW_VjRU or the video can be found on the submission DVD.

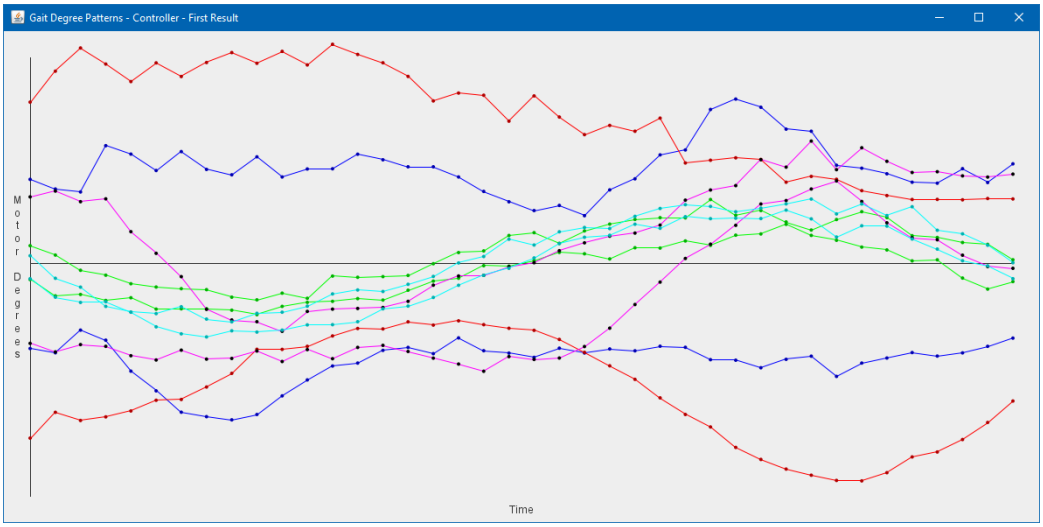
5.1 Distance

The initial criterion for evaluating controllers was the distance the Bioloid travelled using the original equation (Equation 4.1) in Chapter 4.2. The fitness function was improved during these preliminary tests to included a penalty for falling over (Equation 4.2). These tests were performed on the original simulated model before any modifications and increased weight was added. The model weighed 0,58 kg and the scene recorded the final position of the Bioloid after the gait cycled for 5 iterations. The gait needed to cycle several times in order to understand the trajectory of the path it made, as well as making sure the repetition of gait was smooth. The observation during each iteration motivated various changes and improvements to the implementation of the EA. Such changes are explained and referenced back to Chapter 4.

5.1.1 Iteration 1

Even though previous tests were carried out, iteration 1 refers to the first set of valid results worth analysing. The training process only lasted 500 generations as the computers were restarted in the middle of the test. After viewing how the controller had evolved thus far, what changes needed to be made were already clear. Figure 5.1 shows the evolved gait of the best controller.

Figure 5.1: Iteration 1



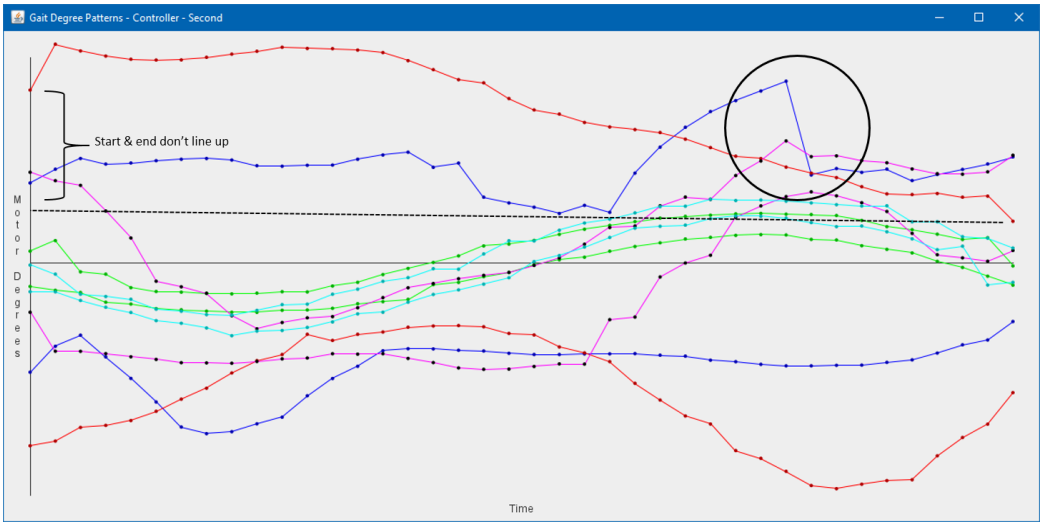
The most evident observation would be the erratic deviations in the motor degrees for each motor. This was caused by only using a single gene mutation and motivated the idea for using grouped mutation mention in Section 4.5. The gait was trained in how straight the Bioloid walked along the x-axis using the fitness function in Section 4.2. The initial thoughts on a gait that produces locomotion in a straight line are that the gait would be symmetrical. By visual inspection of the Bioloid in motion (results video at time 0:40) it was clear that the gait was far from symmetrical. The minor asymmetry of the gait was exploited by the algorithm where one leg would rapidly swing forward that assisted in straightening the position of the Bioloid whilst propelling it forward. According to the fitness function, this first iteration outperformed the original gait, but unfortunately, would not translate well onto the real robot.

5.1.2 Iteration 2

The second iteration attempted to improve smoothness of the motor degree changes by incorporating grouped mutation. Again the training was interrupted around the 500 generation mark but sufficient results had already been captured to allow for an understanding of what to change. In the Figure 5.2 it is clear that the grouped mutation created a smoother gait but another issue could be seen more evidently. The mutations allowed for jumps

between motor degrees and this is seen on the blue line marked out in the figure below. Rapid changes would wear out the motors on the actual robot, so the jumps would need to be minimised. Additionally, the start and end position of the motor degrees on some motors do not line up which effects the continuous motion as the gait cycle is repeated. To overcome this and the jumps between motor degrees, the mutation in the next iteration would account for this by minimising the gap as well as constraining the amount a motor degree could change. The performance of the gait can be seen in the results video at time 0:56. The controller still exploits the asymmetry of the gait.

Figure 5.2: Iteration 2

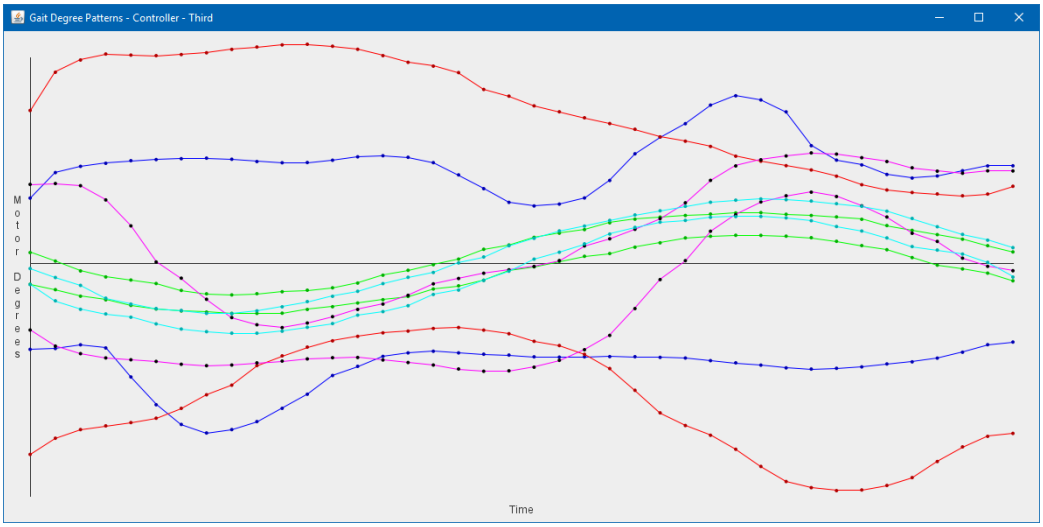


5.1.3 Iteration 3

Once again the training was interrupted and upon investigation, it was found that the computers restart on a daily basis to allow for security updates. This enforced the method to be able to save and load the current state of a population so that with any power cuts or further automatic restarts the evolutionary algorithm would start from a saved state. Again, only 500 generations were completed, but significant improvements to the gait were observed. It took 15 hours to train the 500 generations. Compared to the previous controllers the gait presented in Figure 5.3 shows a substantial improvement to the smoothness as well as continuity of the gait cycle. The swinging motion

of the legs that rapidly advanced the Bioloid forward was reduced and the gait smoother, results video at time 1:13.

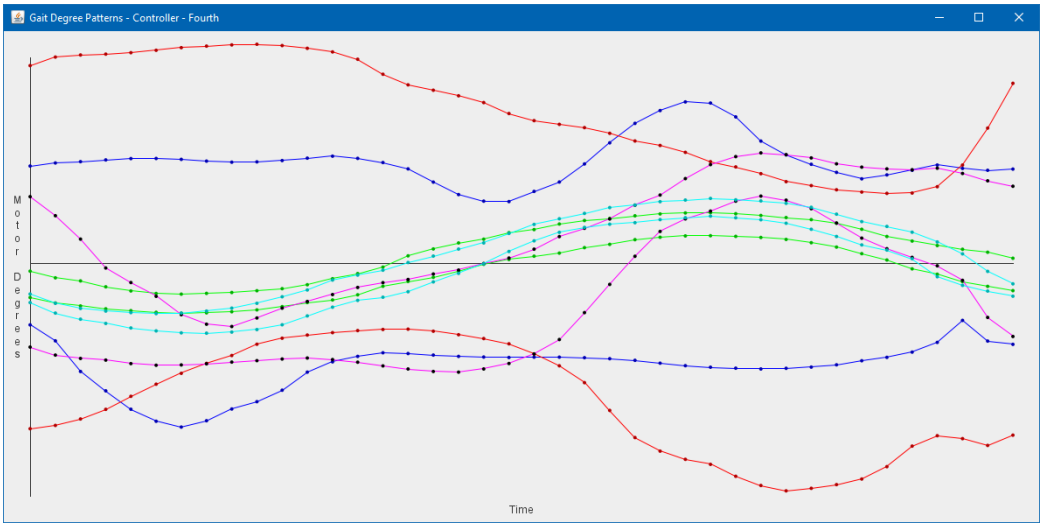
Figure 5.3: Iteration 3



5.1.4 Iteration 4

Up to this point, there had been interruptions in every test to train the controller. Iteration 4 includes minor changes to the constraints and aimed to see the results after all 1000 generations of training. The gait in Figure 5.4 contains the same smooth characteristics of Iteration 3 and evidently attempts to improve the continuity of the gait cycle, results video at time 1:30. Though just from these two graphs it is difficult to determine which one performs better. The results will need to be examined in a more comparative manner. Even though Iterations 3 & 4 shows the graph of the gait becoming smoother their respective videos, along with the previous iterations (results video 0:25-1:47) indicate rapid motor speeds between each motor degree angle.

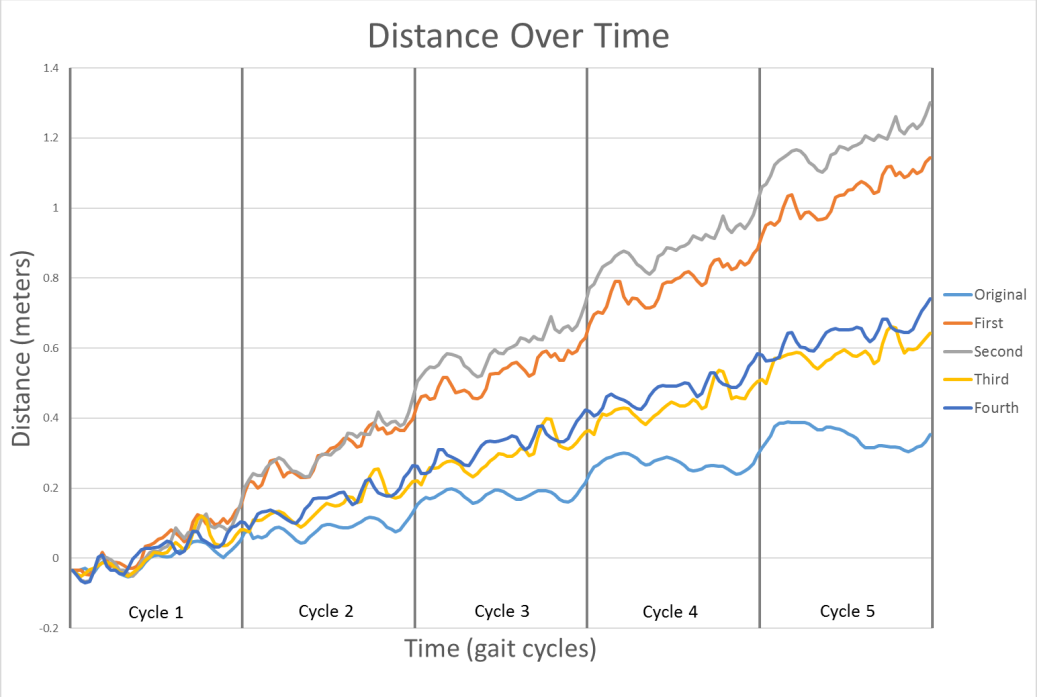
Figure 5.4: Iteration 4



5.1.5 Comparison of Trained Controllers

Since the fitness function evaluates the controllers by the distance covered, the first comparison is simply the distance over time of each controller as displayed in Figure 5.5. Each iteration shows an improvement from the original gait. The first and second iteration covers the furthest distance while iteration three and four displays a less rapid increase. This result may seem to favour the first and second iterations and view them as better though more inspections are made.

Figure 5.5: Distance Over Time



Using position P on the Bioloid's torso, Figure 4.2, that was tracked over multiple gait cycles gave a better indication to the range of movement and the jarring effect the gait had on the Bioloid. The y-displacement, Figure 5.6, shows the swaying motion from left to right and the z-displacement, Figure 5.7, shows how the Bioloid moved up and down. The up and down motion is also caused by the Bioloid leaning forward and backward. What is interesting to see is that the first and second iteration had the larger side-to-side motion while the third and fourth have less of a jolting sway.

Figure 5.6: Y-Displacement

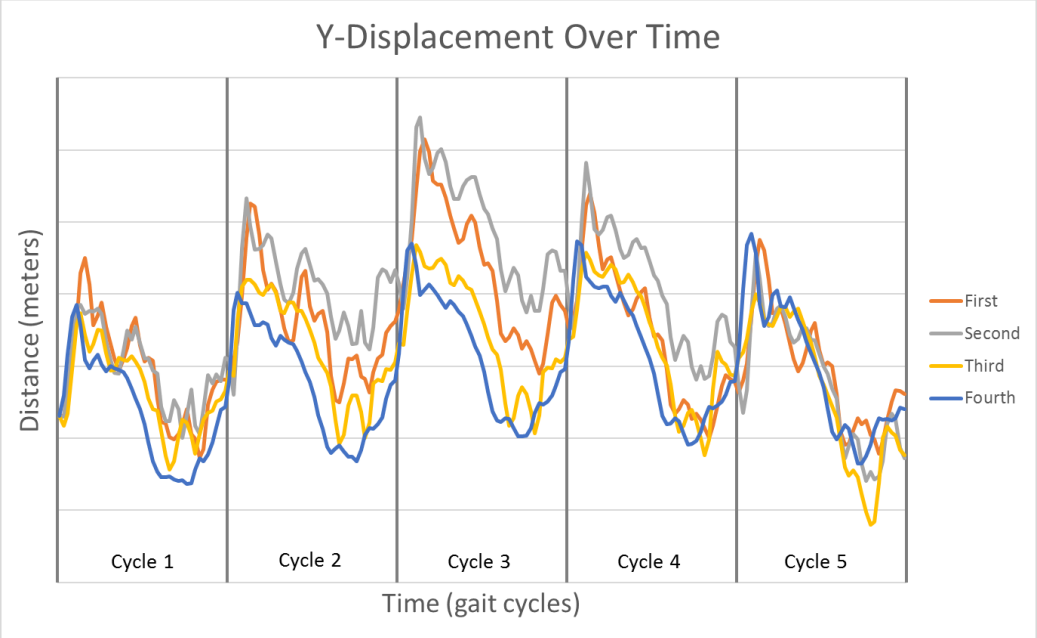
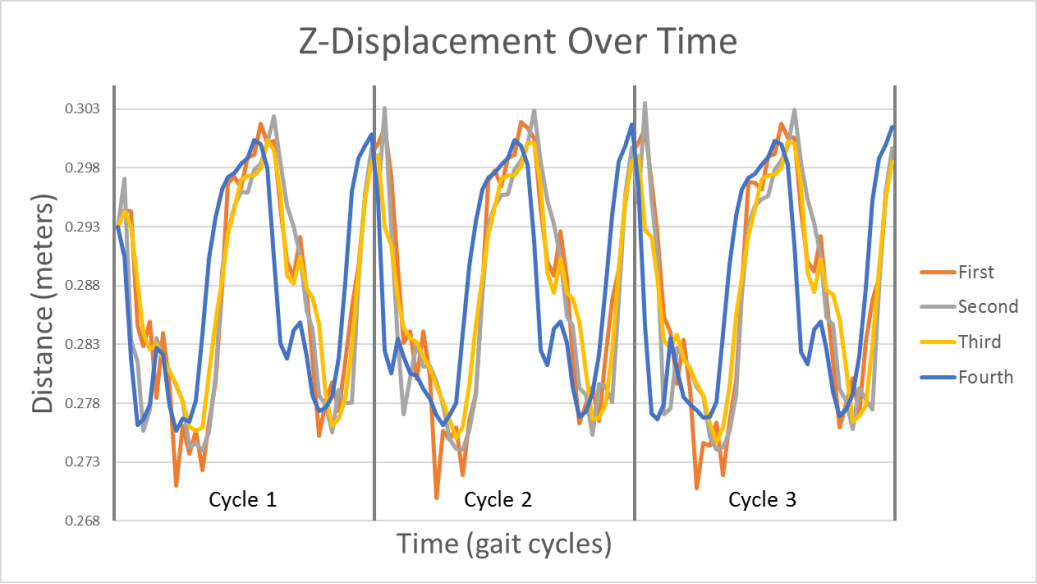


Figure 5.7: Z-Displacement



Similarly by comparing the z-displacement it was shown that the first and second iteration had a slightly greater movement back and forth. Only the first three gait cycles were shown since the z-displacement remains relatively the same throughout the cycles and it provides a clearer graph.

Therefore the first and second iteration may seem to perform better in terms of the distance covered, it was at the cost of rapid movements by swinging limbs and jolting torso. This would translate to a less stable gait on the physical robot. The third and fourth iteration gave fairly similar results but ultimately the fourth iteration covers a greater distance. The controllers are not yet perfect and have room for further improvements nevertheless at this stage, it is evident that the controllers are evolving towards a more optimal solution.

5.2 Stability

When the trained controllers in Section 5.1 were transferred onto the physical Bioloid the errors indicated the reality gap needed to be minimised. The model was inspected and it was found that the weight of the Bioloid in simulation was significantly lighter than the physical one. The actual weight of the physical Bioloid was 1,7 kg while the original model was only 0,58 kg. This meant that the simulated model exploited erratic jerks and swaying motion that when transferred on physical Bioloid would cause it to forcefully fall over, Section 5.3. This section analyses the attempts at minimising the reality gap by increasing the weight of the simulated Bioloid to match the physical as well as including an additional criterion of stability to the fitness function. The literature review focused on the ZMP, Section 2.3.2, to increase stability, unfortunately the model in simulation did not allow for such calculations so an alternative method was used. The results from the controllers in Section 5.1 indicated a rapid motor speed on the Bioloid, which motivated the reduction in motor velocity for the next set of tests.

5.2.1 Increased Weight

The most evident way to minimise the reality gap is to match the weight of the simulation to the physical Bioloid. By increasing the weight in simulation the original gait was unsuccessful in achieving locomotion after a certain

threshold of weight. By slowly incrementing the weight it was found that the original gait could still perform on the simulated Bioloid with a weight of 1,3 kg. Anything higher the EA would get 'trapped' and failed to improve the controller. The trained controller with the increased weight to 1,3 Kg can be seen in the results video at time 1:47. Figure 5.8 shows the original gait compared to the simulation with the increased weight.

Figure 5.8: Original vs Increased Weight

It was evident that the controller with the increased weight had evolved to perform better with regards to gaining distance however it was at the expense of increase movements back and forth, in other words the swaying motion had increased. This was also evident in previous tests, Section 5.1.5.

5.2.2 Accumulative Backwards Sway

Through previous observation on both the simulated model and physical model, it was evident that the Bioloid tilted backwards and forwards where the center of gravity passed the tipping point and caused it to fall over. This

observation lead to an experiment that attempted to minimise the sway. In addition, the weight of the simulated Bioloid needed to be increased, however, the initial accumulative backwards sway (ABS) did not allow for training on the model with full weight of the Bioloid. Therefore, as the ABS was reduced the weight of the model was increased, Figure 5.9 gives an overview of these results. Chapter 4.2 explained the process of applying the ABS as penalty to the fitness function and the results will subsequently be analysed.

Figure 5.9: Accumulative Backwards Sway

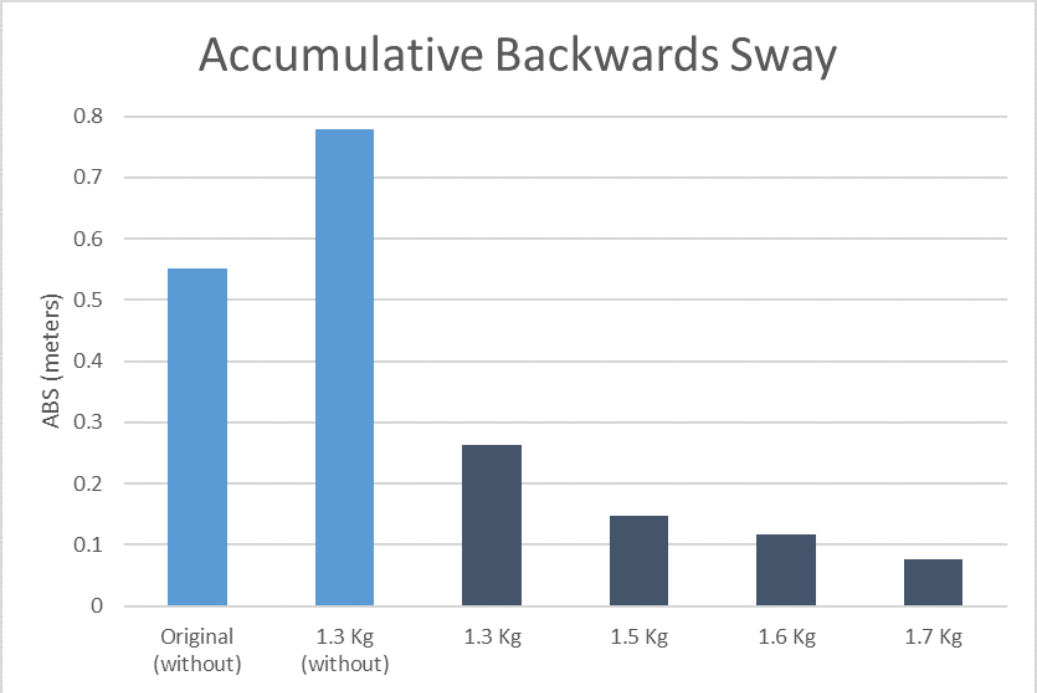


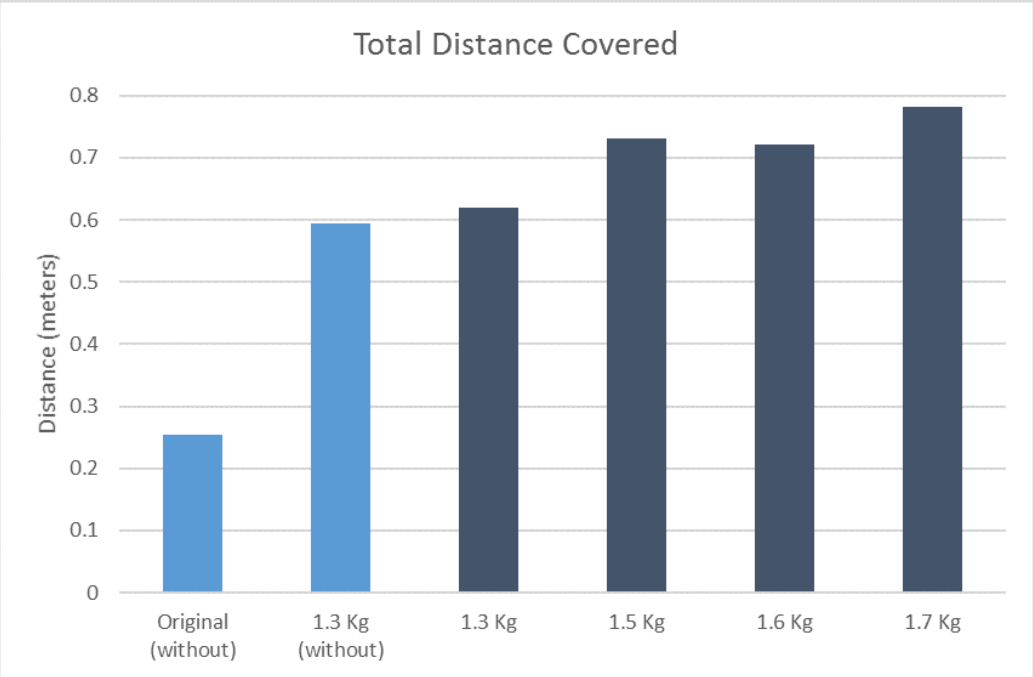
Figure 5.9 includes the original controller and the model with the increased weight to 1,3 Kg *without* the ABS penalty. The test on the 1,3 Kg simulated Bioloid was repeated except it now included the ABS penalty in the fitness function. The first set of result from the training showed a significant reduction in the ABS. This produced a more stable gait where the weight of the simulation could be increased. The next test increased the weight to 1,5 Kg and used the final population during the training of the 1,3 Kg as the initial population, as explained in Chapter 4.1. The process was repeated until the weight reached 1,7 Kg, the weight of the physical Bioloid.

As seen in Figure 5.9, the ABS continuously gets minimised. Furthermore, this is not only evident in the analysis in the data but also the stabilising effect of the ABS can be seen visually on the trained controllers. The result video, from time 1:47-3:16, displays the evolution process of improving the stability as the weight is increased.

5.2.3 Distance Covered

The distance covered still remained the primary objective during the evaluation and the ABS penalty aimed to reduce the negative characteristics of the swaying motion of the gait. Figure 5.10 displays the distance covered by the evolved controllers through the various tests.

Figure 5.10: Distance Covered with ABS Penalty



It is evident that the distance gained is becoming greater throughout the different tests, though the rate is slowly decreasing.

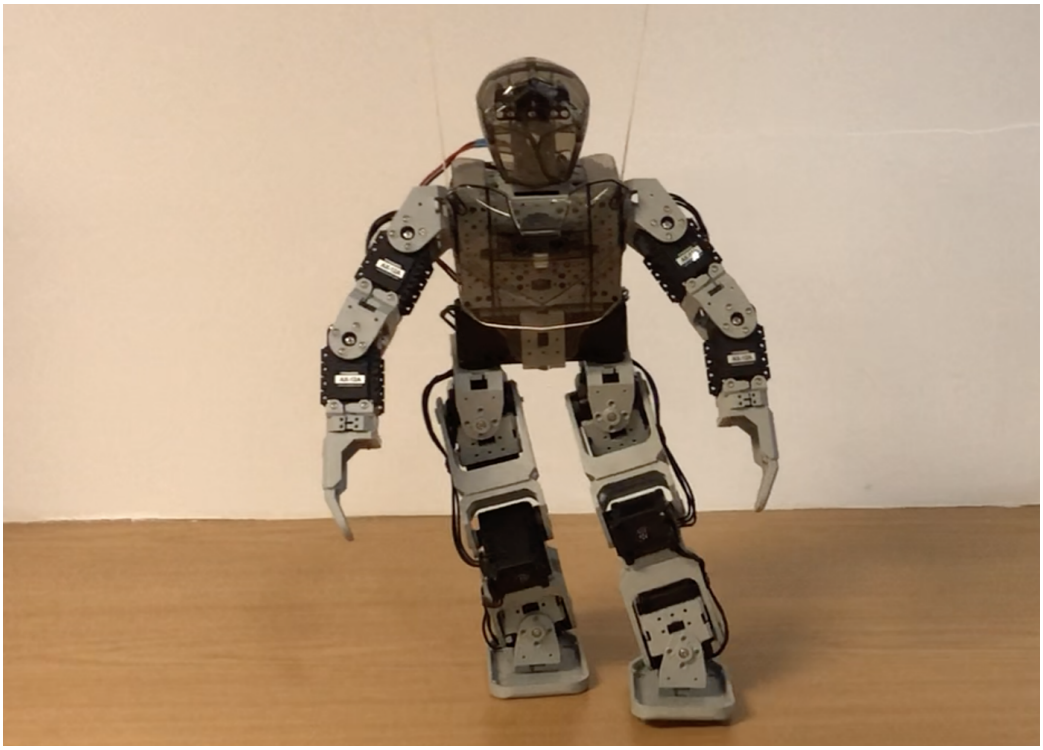
Another interesting observation during the first set of generation training with an increased weight was how most of the initial population would fall

over and by the final generation the entire population was converging towards the solution with minimal individuals falling over.

5.3 Physical Bioloid

After the controllers were trained in simulation they were transferred onto the physical Bioloid. Due to the nature of the physical Bioloid, information about how the controllers performed could only be achieved through observation. The results video from time 3:16 displays the various controllers that were tested on the physical Bioloid. The program for the CM530 was only refined after the inclusion of stability to the fitness function, therefore, none of the preliminary results of the different iterations in Section 5.1 are seen in the video. In addition, the drastic jarring motion in the earlier iterations could damage the motor and hence did not get transferred onto the physical Bioloid. Figure 5.11 shows the Bioloid performing a trained controller.

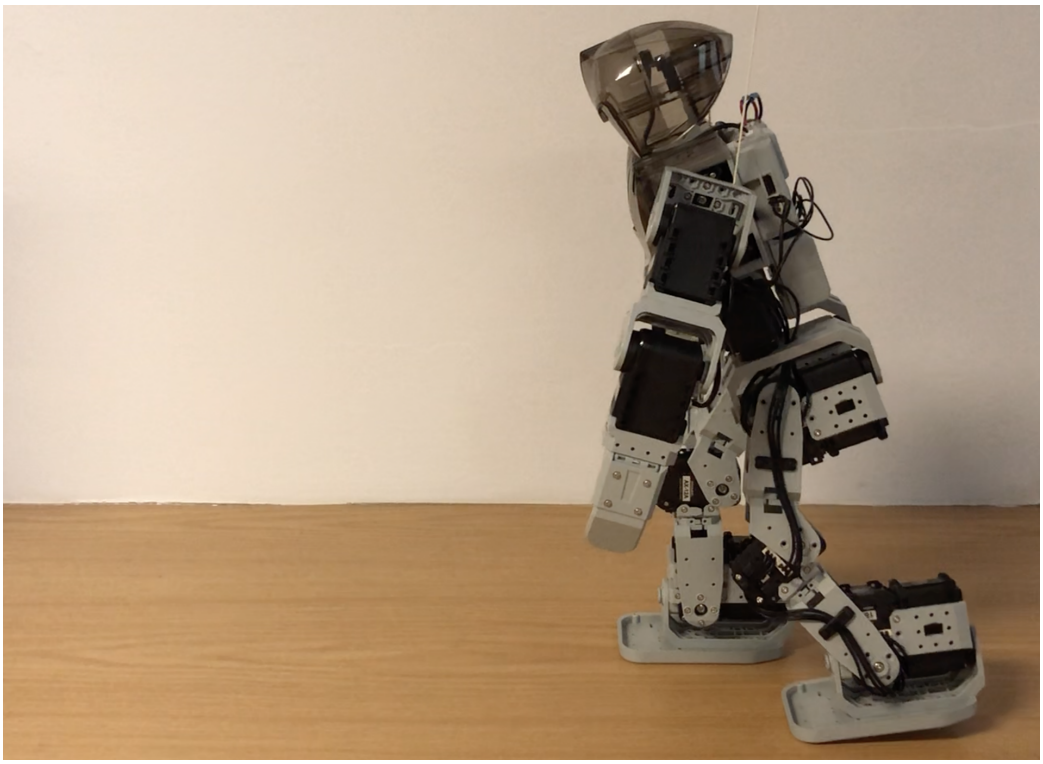
Figure 5.11: Bioloid Front



5.3.1 Reality Gap

Figures 5.11, 5.12 as well as the results video shows the Bioloid supported by two chords, this was to protect the Bioloid from damage when it fell over. The simulation of the Bioloid indicated that the controller achieved successful locomotion. Unfortunately, none of the trained controllers were able to walk without support. The support also helped display the gait if it was able to move without falling over. The difference between simulation and the real environment can be summarised as the reality gap problem, Section 2.2. Refinements need to be done to the simulated model and the physical model in order to minimise any differences between them.

Figure 5.12: Bioloid Side



5.3.2 Motor Speeds

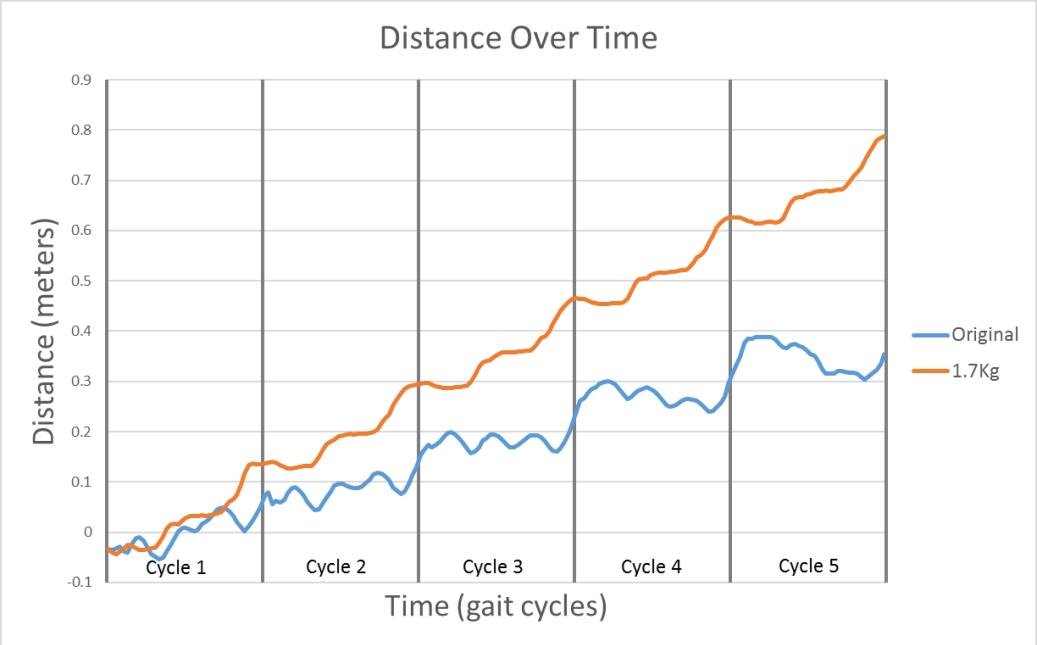
The original controller had a shaky gait which was caused by the motor speeds. The gait would change between motor degrees at maximum velocity

and once the position of the next motor degree was reached the controller would wait for the next set of commands which had a slight delay. The motor velocity on the controller was reduced in an attempt to smooth the gait. Figure 5.12 shows a side view of the results video at time 5:06 - 5:34 comparing the original gait to the final 1,7 Kg controller. The sound of the video was included to support the contrast of the two controllers. By evidence of the video the 1,7 Kg controller looks and sounds smoother.

5.4 Final Controller

Through the various iterations and improvements to the EA a final controller had been evolved. A comparative study between the original gait and the evolved gait aims to show the effects the EA had on the original gait during the evolutionary process. Both the original gait and final evolved gait can be seen performed on the simulated Bioloid in the results video at time 0:22 and 2:59 respectively. The first optimisation criterion is distance where there is a significant improvement, Figure 5.13 gives a representation of the distance covered as the gaits are cycled several times.

Figure 5.13: Distance Over Time

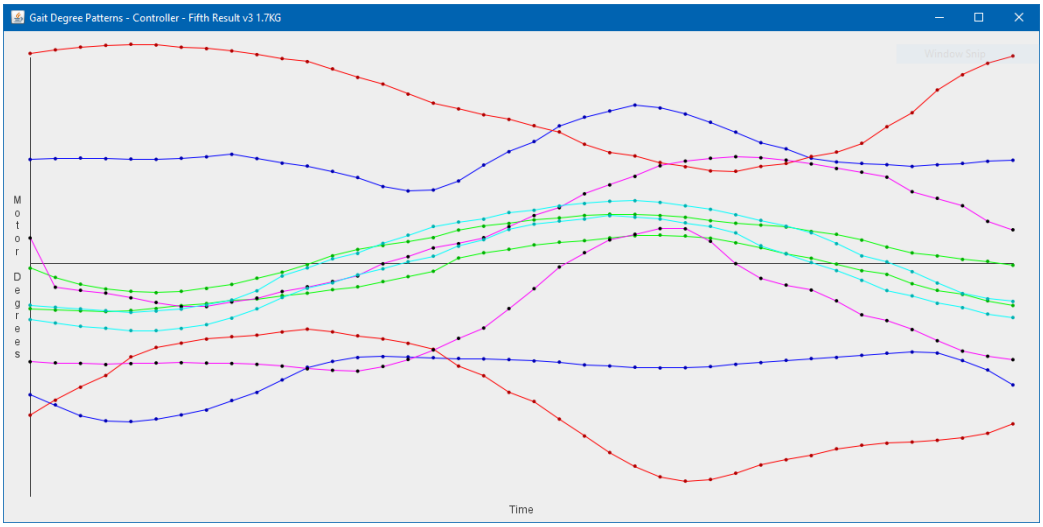


Not only does Figure 5.13 indicate that the final controller travels further it can been seen that the distance increases at a smoother and more consistent rate. This leads to the next optimisation criterion which was stability. The ABS gives a representation of stability and the ABS of the final controller is seen in Figure 5.14.

Figure 5.14: Distance Covered and ABS

Figure 5.14 also shows the total distance covered by the controllers which provides a summary of the results. The EA was shown to maximise the distance covered and minimise the ABS to improve stability, thus improving the original gait based on the two optimisation criteria. The graphical representation of the motor degrees of the final controller is shown in Figure 5.15, which will help understand what changes have been made to the original gait and throughout the training process.

Figure 5.15: Final Gait



The beginning of the gait cycle matches the end which makes the repetition of the gait smoother. Similarly, the constraints to the mutations and crossovers ensured no drastic changes in the motor degrees.

5.5 Conclusion

The first iterations of the results indicated certain changes needed to be made to the EA that trained the controllers. These changes were mainly to the reproduction operators in order to evolve and produce a smoother gait. The instability of the controller was observed and the results indicate an improvement. To quantify the stability of a gait the ABS is calculated to be added as a penalty to the fitness function. The comparison between the original gait and the final evolved controller showed a clear improvement to both the distance covered and increased stability. Unfortunately the reality gap would need to be minimised in order for the final controller to achieve successful locomotion on the physical Bioloid.

Chapter 6

Conclusion

The project saw research into the field of evolutionary robotics that followed principles of evolutionary algorithms. The aim was to apply ER techniques to the optimisation of the walking gait of a robot, specifically the Robotis Bioloid Premium. This final chapter serves to conclude the research and results achieved during development. Section 6.1 reiterates the goals and what was aimed to be achieved which follows into an overview of the development process in the attempt to attain successful results, Section 6.2. The results are summarised in Section 6.3 which motivates potential future development, Section 6.4.

6.1 Project Overview

The project aimed to optimise the walking gait on the Robotis Bioloid Premium. An existing gait needed to be found and improved based on some criteria. Previous success in using evolutionary robotics focused methodology on optimising the gait by using evolutionary algorithms. This meant an appropriate controller needed to be designed to represent the gait. The process to evolve the controller would need the assistance of a simulation that modeled the Bioloid. Once a controller had been evolved in simulation it needed to be transferred onto the physical Bioloid. A brief attempt was made to evolve a controller from a random initialisation of the gait. This was beyond the project scope and was soon discovered to be considerably more complex.

6.2 Project Development

The development of the algorithm to optimise the walking gait followed principles of a Genetic Algorithm, where an individual was the controller representing the gait. The structure of the gait was explored which lead to representation of the gait as a two-dimensional matrix of motor degrees for each motor (joint) in the Bioloid. The operations in the EA were continuously refined as results motivated certain changes. The reproduction operators were improved to create smoother changes in the gait cycle and initialisation saw the original gait being mutated to explore the solution space. The fitness function focused on the optimisation criteria of distance and stability. Where, distance was simply the Euclidean distance and stability seen as the accumulative backwards sway, a term coined throughout the experiments. The time taken to evaluate a single controller meant that the training process needed to be drastically reduced in order to run multiple tests and experiments throughout the development of the EA. The solution saw a network of computers that shared the evaluation of a population of candidate controllers. The trained controllers would need to be transferred onto the physical Bioloid where the original CM530 controller on the Bioloid was overwritten with the gait of an evolved controller. The reality gap indicated there was a difference in the simulated and physical Bioloid. Further development would be needed to improve the simulation and training.

6.3 Results

Throughout the development of the evolutionary algorithm the results achieved were used to improve the implementation of the algorithm. Analysis of the graphical representation of the gait during the earlier iterations motivated changes to the reproductive operators. Improvements helped to smooth changes in the motor degrees on the controller. The observation of the amount the Bioloid tilted forwards and backwards led to the development of the accumulative backwards sway (ABS) in an effort to improve stability. The results saw the swaying motion reduce significantly as the ABS penalised the characteristics of the swaying motion in the gait. Unfortunately, the evolved controllers were unsuccessful in achieving locomotion on the physical Bioloid. The efforts to reduce the reality gap produced better results, but the physical Bioloid would still topple over without any support.

The focus was then directed on the success of the EA optimising the gait in the simulation. Where, the final evolved controller travelled further with a reduced ABS, the measurement used to quantify stability, compared to the original gait.

6.4 Future Development

Even though the evolutionary algorithm created proved successful in improving the original gait in simulation, when transferred onto the physical Bioloid there was discrepancy in the results. Many factors affected the difference between simulation and the real environment which offers opportunity for further improvements and development. During implementation the reality gap was reduced, though, significant improvements can still be made. One such improvement would be to simulate the Bioloid more accurately, ensuring that each component in the model functions the way it would in the real world. The complexity of functionality of the gait was considerably reduced into a series of motor degree changes with a constant velocity. However, the velocity of each motor would most likely be different and change throughout the cycle, contributing to a more fluid gait. The studied gait only included the motion of the motors in the Bioloid's legs, where also the swinging motion of arms could be factored in to increase stability. The most prospective future development would be creating an random initialisation of gait and the Bioloid learns to walk instead of optimising an existing gait.

6.5 Summary

The principles of Evolutionary Robotics were applied to the development of a controller that represented the walking gait of the Robotis Bioloid Premium. The controller was evolved in simulation by the use of an Evolutionary Algorithm and then transferred onto the physical Bioloid. The results saw success in optimising the original gait during simulation in terms of the distance covered and stability. However, the gait failed to perform accurately in the real-world environment. Future development could see improved design of the physical robot gait, as well as developing the algorithm that learns to create the walking gait from a random initialised population. Ultimately, the project saw success in optimising the walking gait of a humanoid robot.

Bibliography

[1] Bongard. Taking a biological inspired approach to the design of autonomous, adaptive machines. *Communications of the ACM*, 56(8):74–83, 2013.

[2] J. Bongard. Morphological change in machines accelerates the evolution of robust behavior. *Proceedings of the National Academy of Sciences*, 108(4):1234–1239, 2011.

[3] G. Correia. Bioloid-java-simulator. https://github.com/LAB08-SBC/Bioloid-Java-Simulator, 2015.

[4] B. Dynamics. Atlas. https://www.bostondynamics.com/atlas. Accessed: 2019-04-20.

[5] R. Elio, J. Hoover, I. Nikolaidis, M. Salavatipour, L. Stewart, and K. Wong. About computing science research methodology, 2011.

[6] A. P. Engelbrecht. *Computational intelligence: an introduction*. John Wiley & Sons, 2007.

[7] D. Gong, J. Yan, and G. Zuo. A review of gait optimization based on evolutionary computation. *Applied Computational Intelligence and Soft Computing*, 2010, 2010.

[8] K. Hirai, M. Hirose, Y. Haikawa, and T. Takenaka. The development of honda humanoid robot. In *Proceedings. 1998 IEEE International Conference on Robotics and Automation (Cat. No. 98CH36146)*, volume 2, pages 1321–1326. IEEE, 1998.

[9] J. K. Hodgins. Biped gait transitions. In *Proceedings. 1991 IEEE International Conference on Robotics and Automation*, pages 2092–2097. IEEE, 1991.

[10] A. Howard and N. Koenig. Webots. https://cyberbotics.com/#webots, 2019. Accessed: 2019-07-29.

[11] K. Ichiro et al. Development of waseda robot. *Humanoid Robotics Institute, Waseda University*.

[12] N. Jakobi, P. Husbands, and I. Harvey. Noise and the reality gap: The use of simulation in evolutionary robotics. In *European Conference on Artificial Life*, pages 704–720. Springer, 1995.

[13] K. Kaneko, F. Kanehiro, S. Kajita, K. Yokoyama, K. Akachi, T. Kawasaki, S. Ota, and T. Isozumi. Design of prototype humanoid robotics platform for hrp. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, volume 3, pages 2431–2436. IEEE, 2002.

[14] M. Mitchell. *An introduction to genetic algorithms*. 1998.

[15] S. Nolfi, D. Floreano, and D. D. Floreano. *Evolutionary robotics: The biology, intelligence, and technology of self-organizing machines*. MIT press, 2000.

[16] Oracle. Java. https://www.java.com/en/, 2019. Accessed: 2019-07-29.

[17] I.-W. Park, J.-Y. Kim, J. Lee, and J.-H. Oh. Online free walking trajectory generation for biped humanoid robot khr-3 (hubo). In *Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006.*, pages 1231–1236. IEEE, 2006.

[18] J. Perry and M. Burnfield. Gait analysis: normal and pathological function. *Journal of Sports Science and Medicine*, 9(2):353, 2010.

[19] J. Pratt, P. Dilworth, and G. Pratt. Virtual model control of a bipedal walking robot. In *Proceedings of International Conference on Robotics and Automation*, volume 1, pages 193–198. IEEE, 1997.

[20] C. J. Pretorius, M. C. du Plessis, and J. W. Gonsalves. Evolutionary robotics applied to hexapod locomotion: a comparative study of simulation techniques. *Journal of Intelligent & Robotic Systems*, pages 1–23, 2019.

[21] PUC-Rio. Lua. https://www.lua.org/, 2019. Accessed: 2019-07-29.

[22] T. Reil and P. Husbands. Evolution of central pattern generators for bipedal walking in a real-time physics environment. *IEEE Transactions on Evolutionary Computation*, 6(2):159–168, 2002.

[23] C. Robotics. Gazebo. http://gazebosim.org/, 2019. Accessed: 2019-07-29.

[24] C. Robotics. V-rep pro edu. http://www.coppeliarobotics.com/, 2019. Accessed: 2019-07-29.

[25] Robotis. Robotis bioloid premium e-manual. http://emanual.robotis.com/docs/en/edu/bioloid/premium/. Accessed: 2019-04-19.

[26] F. Silva, L. Correia, and A. L. Christensen. Evolutionary robotics. *Scholarpedia*, (7), 2016.

[27] T. Stöckel, R. Jacksteit, M. Behrens, R. Skripitz, R. Bader, and A. Mau-Moeller. The mental representation of the human gait in young and older adults. *Frontiers in psychology*, 6:943, 2015.

[28] A. Tettamanzi and M. Tomassini. *Soft computing: integrating evolutionary, neural, and fuzzy systems*. Springer Science & Business Media, 2013.

[29] M. Vukobratović and B. Borovac. Zero-moment point—thirty five years of its life. *International journal of humanoid robotics*, 1(01):157–173, 2004.

[30] G. W. Woodford, M. C. Du Plessis, and C. J. Pretorius. Evolving snake robot controllers using artificial neural networks as an alternative to a physics-based simulator. In *2015 IEEE Symposium Series on Computational Intelligence*, pages 267–274. IEEE, 2015.

[31] J. C. Zagal and J. Ruiz-Del-Solar. Combining simulation and reality in evolutionary robotics. *Journal of Intelligent and Robotic Systems*, 50(1):19–39, 2007.

[32] R. Zhang, P. Vadakkepat, and C.-M. Chew. An evolutionary algorithm for trajectory based gait generation of biped robot. In *Proceedings of the International Conference on Computational Intelligence, Robotics and Autonomous Systems*, volume 144, 2003.