2023

Effects of Morphology on Genetic Assimilation of Learned Behavior

Natalie L. Tolley
University of Vermont

Recommended Citation

Tolley, Natalie L., "Effects of Morphology on Genetic Assimilation of Learned Behavior" (2023). *Graduate College Dissertations and Theses*. 1771.
https://scholarworks.uvm.edu/graddis/1771

Effects of Morphology on Genetic Assimilation of Learned Behavior

A Thesis Presented

by

Natalie L. Tolley

to

The Faculty of the Graduate College

of

The University of Vermont

In Partial Fulfillment of the Requirements
for the Degree of Master of Science
Specializing in Computer Science

August, 2023

Defense Date: July 20, 2023
Thesis Examination Committee:

Joshua Bongard, Ph.D., Advisor
Dryver Huston, Ph.D., Chairperson
Nicholas Cheney, Ph.D.
Cynthia Forehand, Ph.D., Dean of the Graduate College

ABSTRACT

The Baldwin effect is an evolutionary theory regarding the assimilation of ontogenetic changes into a population's genome via selection pressure to entrench beneficial phenotypes discovered through learning. In evolutionary computation, the incorporation of learning into non-embodied agents allows them to navigate otherwise rough fitness landscapes by allowing for local exploration at particular points in that landscape. Prior work investigating the specific mechanisms by which learned behavior is genetically assimilated is almost entirely limited to non-situated, non-embodied simulations such as bitstring manipulation. However, recent research has demonstrated that genetic assimilation can be observed in embodied agents. Learning more about the ways embodiment may affect the mechanisms of genetic assimilation can help us better understand how learning can affect evolution and enhance our design of evolved, learning, embodied systems.

To accomplish this, we co-evolve the initial values and learning rules for each synapse in the controlling neural network of three different robots to investigate the impact of morphology on the Baldwin Effect. We found that the different morphologies tested were capable of genetic assimilation within the evolutionary timespans provided and that each morphology exhibited significant differences in the amounts of genetic assimilation they were capable of. These differences were due entirely to different behaviors between morphologies, while the rate of genetic assimilation due to evolution of non-learning synaptic weights was constant regardless of morphology.

I dedicate this thesis to my cat, Chickadee, who has never learned a thing.

ACKNOWLEDGEMENTS

I would like to express my deep gratitude to my advisor, Josh Bongard, for his support, patience, and mentorship throughout my research. Every time I felt that my research had hit a dead end, he managed to turn the problem over until I saw the possibilities in what had seemed to be obstacles. Without his help, I never would have completed this research. I would also like to thank the entire Morphology, Evolution, and Cognition Lab for their questions, suggestions, and interest in my work.

Lastly, I want to thank my family, especially my parents Kirsten and George Tolley, for their support and encouragement to pursue a graduate degree.

Computations were performed, in part, on the Vermont Advanced Computing Center.

Table of Contents

CHAPTER 1

INTRODUCTION AND BACKGROUND

1.1 INTRODUCTION

Design of embodied agents, also known as robots, is a complex task. Any given robot must not only be equipped with a body capable of performing its duties, but a control policy capable of manipulating this body even in adverse or unforeseen circumstances. Given that robots are most commonly used in scenarios humans find dangerous, messy, or difficult, a powerful control policy is required to ensure that tasks are successfully completed. Because the creation of such a controller is such a complex task, evolutionary algorithms are often employed to automate the design process.

The field of evolutionary robotics centers around the use of evolutionary algorithms to automate the development of robotic systems, drawing inspiration from the ways evolution acts on biological organisms. Evolutionary algorithms are an ideal choice for the design of robotic control and morphology, where parameters are difficult to optimize due to the highly nonlinear nature of the problems at hand. However, evolutionary algorithms suffer from their own shortcomings in certain highly complex design spaces, many of which can be addressed through the introduction of learning capabilities to evolved agents. Recent research has shown that exploiting a feature of evolution known as the Baldwin effect, which allows evolution to tune where and how learning occurs, can produce significantly higher-fitness solutions than approaches that cannot exploit this effect. Understanding how the Baldwin effect operates in embodied systems can help us better exploit it in future research to produce more useful and capable machines, as well as to understand the ways in which lifetime learning interacts with evolution in organisms.

1.2 Artificial Neural Networks

Control of robotic systems often takes the form of an artificial neural network, a form of information processing which takes inspiration from the function of organic nervous systems. At the most basic level, they take information in some numerical form and transform it via several parallel and serial operations into a numerical output. When used to solve problems, neural control policies can be open loop or closed loop. An open loop controller processes information composed of discrete instances that may have no relation, like in image recognition. A closed-loop controller describes scenarios where the output of the network affects the information that comes in next, like the controller of a robot.

There are two components to a neural network: the neurons and the synapses. The function of a neuron is to sum incoming values from other neurons it is connected to, perform a single mathematical operation on this sum, and then pass the output along its outgoing connections to other neurons. The synapses are responsible for connecting neurons to each other, and modulate the signals they transmit. Each synapse has an associated synaptic weight, which determines how much it inhibits or excites the output of its presynaptic neuron. A synaptic weight of greater magnitude will not dampen the signal at all, while a synaptic weight of 0 will nullify the signal completely, and a negative synaptic weight will flip the signal of its presynaptic neuron. The input to a postsynaptic neuron is the dot product of the output of the presynaptic neurons connected to it and the weights of the synaptic connections those outputs are being transmitted over [1]. A given neuron n_j with presynaptic neurons $< n_1, n_2, ..., n_i >$, each with output o_i transmitting along synaptic connections $< s_{1j}, s_{2j}, ..., s_{ij} >$ with weight w_{ij} will have an input m_j demonstrated by the following equation:

$$m_j =< n_1, n_2, ..., n_j > \cdot < w_{1j}, w_{2j}, ..., w_{ij} >$$

In order for a neural network to produce useful output, the weights of these synaptic connections must be modified and tuned. Determining the ideal synaptic weights so that a network exhibits the desired properties is non-intuitive, so the process is automated via a gradient descent algorithm. To tune a synapse, the neural network undergoes a training phase where the ideal output for a given input is already known, and the network can therefore be judged by how accurate it is. The responsibility for deviation between the desired output and the network's output can then be propagated backwards across the synapses, assigned proportional to their calculated role in producing the result. As synapses are adjusted, the neural network travels across a "gradient" of performance, moving further down the slope as it reduces error.

However, there exist applications for which error cannot be back-propagated, such as our robot example. When we have no idea what the ideal control policy looks like, no error can be calculated and no gradient towards a desired output can be formed. Further, in systems where ANNs are required to embody a highly-nonlinear transformation of input into output, the impact of any one synapse at a given moment of time is difficult to calculate, making attribution of the final outcome to any individual synaptic weight difficult. This is known as the credit assignment problem [2]. In cases such as these, unsupervised gradient-free methods such as evolutionary algorithms become useful [3].

1.3 Evolutionary Algorithms

Evolutionary algorithms are a stochastic method of searching a problem's solution space that takes inspiration from biological evolution. They are particularly useful for optimisation problems where the best solution is not known, or where the mapping from input to output is too non-linear to predict how the system should be changed to improve its behavior, and so are ideal for designing the controllers for embodied agents.

1.3.1 Evolutionary Algorithm Cycle

Unlike other supervised learning methods, evolutionary algorithms work with no optimal solution to measure results against. Instead, a group of solutions (also referred to as a population of individuals) is iterated through a cycle of evaluation, selection, and mutation. During evaluation, individuals are evaluated and given a quantifiable measure of how well they performed, called their fitness. This fitness is used to compare different individuals against each other, to determine which performed the best in the selection phase. In selection, individuals with lower fitness are removed from the population while those whose fitness is relatively higher remain. In mutation, remaining individuals are copied and altered in some way to produce new solutions. The new individuals are then evaluated, and the cycle repeats.

1.3.2 Genotype and Phenotype

We use the term "genotype" to refer to the variable values making up an individual solution. The genotype typically takes the form of a vector of values, and dictates the way the individual performs when evaluated in ways similar to how a biological genome affects an animal's features. The phenotype of the individual is the behavior displayed when the genome is applied to the problem space as a solution. A neural network controller for a robot typically has a genotype made up of specific synaptic weights, while the phenotype would be the movement that the robot produces when simulated with that set of synaptic weights. While the genotype is passed from parent to offspring directly, only altered by mutation, the phenotype, which is only passed indirectly via expression of similar genotypes in similar environments, is the way by which fitness can be measured.

1.3.3 Fitness Landscape

The fitness landscape of a particular problem refers to the mapping of genomes to their respective fitness values. The term refers to the way that variations in fitness between solutions create "peaks" and "valleys" of performance. Evolutionary algorithms generate solutions randomly within solution-space, and then "climb" the slopes of higher fitness by making incremental steps through this space with mutation, with the goal of reaching a globally-optimal solution, but in practice often becoming trapped in local optima. It is important to note that while local "peaks" of fitness can be identified within solution-space, it is often impossible to determine whether a given solution is truly optimal or simply optimal for the area covered by iterations of the algorithm.

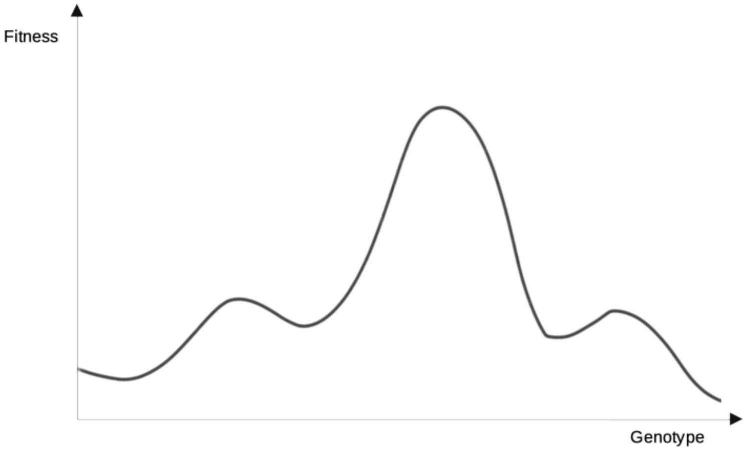

Figure 1.1: A plot demonstrating the relationship between genotype and fitness in a fitness landscape. Most genotypes have many dimensions, and cannot be flattened to a simple 2D plot, though principles demonstrated in simplified plots apply to higher-dimensional landscapes.

Due to the lack of information about the shape of the fitness landscape within the algorithm itself, there are certain fitness landscapes which are difficult for evolutionary algorithms to traverse and find solutions in: landscapes with too many peaks and valleys may trap evolutionary algorithms in local optima early in the process, and are called "rugged". In addition, barren landscapes with only a few, very localized points of fitness are intractable due to the lack of a slope for evolution by mutation to ascend. Due to the exhaustive search required to solve these landscapes, they are also referred to as "needle in the haystack" landscapes.

Current approaches to the problem of rugged fitness landscapes revolve around encouraging diversity in a population to avoid early convergence on a sub-optimal genotype, through techniques such as age-fitness Pareto optimization [4] or selecting for novelty over fitness [5]. Fitness landscapes with isolated high-performance genotypes, meanwhile, may be "smoothed" with the addition of learning, which causes some component of the agent's phenotype to change over its lifetime. This transforms it from a single point in the fitness landscape to a localized trajectory, and thus increases its chances of stumbling onto a fitness gradient.

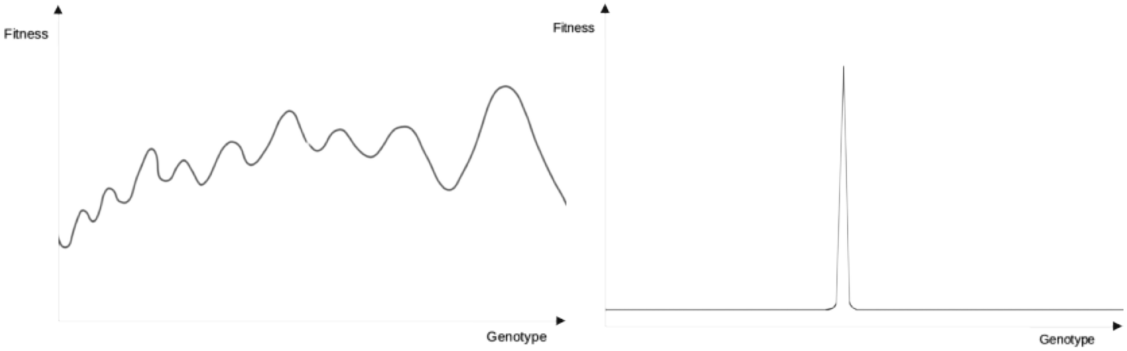

Figure 1.2: Difficult-to-traverse landscapes may confound simpler evolutionary algorithms. Especially rugged landscapes (left) have many local optima that result in very early convergence without the ability to explore further. Needle in the haystack landscapes (right) have little to no slope for mutation to climb, and locating the peak through mutation becomes an intractable exhaustive-search problem at even moderate problem-space complexity.

1.4 LEARNING IN NEURAL NETWORKS

In this work we use 'learning' to specifically refer to the modifications a neural network undergoes while actively being simulated, as opposed to the generation-level modifications made by the genetic algorithm. The term "Evolved Plastic Artificial Neural Network" (EPANN), coined by Soltoggio *et al.*, refers to neural networks with evolved parameters that are also capable of learning as the information they receive changes [6]. In their review of current EPANN learning techniques, Soltoggio *et al.* identified Hebbian learning as being of specific interest, due to its roots in theories of biological neural function and capacity for entirely unsupervised learning.

1.4.1 Hebbian Learning

Hebbian learning is a type of unsupervised neural-network training that follows principles of synaptic connectivity first proposed by Donald Hebb, who theorized that synaptic connections would grow in strength when the presynaptic neuron's firing was shortly followed by the postsynaptic neuron's activation [7]. Willshaw and Dayan proposed a modified Hebbian theory featuring a set of three different learning rules which would allow for the depression of synaptic connections under various conditions [8]. Further investigation of the role of depressive synaptic modifications by Miller and Mackay showed that the type of depression mattered as well. Synaptic constraints that decreased by a constant amount tended to 'sharpen' networks down to the most highly-correlated connections, while constraints that decreased proportional to the current synaptic weight more evenly spread the signalling of the network [9].

Floreano and Mondada used these principles when designing learning rules for a physical robot which could navigate a very simple track. The robot's neural network, which was composed of 27 synapses, could modify the strength of those synapses according one out of four possible equations. Each equation rewarded a different kind of relationship between neurons: firing together, exclusively pre- or post- synaptic activation, and modification proportional to the neuron's similarity. The type of equation attached to a given synapse, as well as the learning rate of the synapse, were evolved parameters. Their findings demonstrated the real-life utility of Hebbian learning systems, as well as an example of dynamic stability - though the controller was in a constant state of change, the behavior of the robot remained consistent [10].

With more modern evolutionary strategies capable of scaling development to much larger parameter sets [11], Hebbian learning rules can now be applied to controllers for much larger networks. Recent examples of Hebbian-inspired controllers in embodied agents find that not only can a properly-developed Hebbian parameter set fully adjust randomly-generated networks, networks operating under Hebbian learning rules are much more adaptive to novel scenarios than non-learning networks [12] [13]. Additionally, the Hebbian parameter sets could reassert full capability after widespread perturbation to the network, and once control is asserted, networks' learning could be switched off with minimal loss of function. As opposed to earlier work, these Hebbian learning rules operated on an *ABCD* Hebbian model, where each synapse has its own set of tuned parameters governing how it changes.

1.5 THE BALDWIN EFFECT

The Baldwin effect describes a theory in evolutionary science by which changes in individual phenotypes in a population can drive genetic assimilation of those changes into the genotype [14]. Individuals who discover beneficial phenotypes instigate an evolutionary pressure to perform that phenotype in their population, until the desired trait is assimilated into the population's genome as an inherent, rather than acquired, property. This theory was expanded upon by Waddington [15] to describe how genetic assimilation might occur in a population. Phenotypic variation due to environmental factors must be possible, along with the eventual transfer from external inducement of change to internal. The population must also experience genetic variation and a selection pressure for the phenotype in question [15]. The two key factors outlined by Baldwin and Waddington were genetic assimilation, the encoding of a previously ontogenetically acquired trait into the genome, and canalization, the elimination of environmental interaction as a factor in the trait's expression.

A classic example of this phenomenon can be observed in ostriches, who are born with callus tissue on the bottoms of their feet. Ostriches, who run on abrasive terrain their entire lives, require these calluses to be able to move about without injuring themselves. The development of callus tissue when skin is abraded is encoded into the genome of an ostrich, a beneficial phenotypic change. The evolutionary pressure to develop calluses as quickly as possible eventually results in the genetic assimilation of the trait [16].

1.5.1 Computational Models

The Baldwin effect was first demonstrated in a computational model in 1987, by Hinton and Nowlan [17]. In their research, they demonstrated how a bitstring-maximisation problem could be solved with evolutionary algorithms even if only the maximum value bitstring was assigned a non-zero fitness, forming an extreme 'needle in the haystack' fitness landscape. The addition of learning allowed what would have been non-viable genomes which were close to the solution the possibility to stumble upon the correct string during their lifetimes. By assigning fitness proportional to the amount of time an individual took to locate the solution string, the 'needle' was widened into a climbable slope, with the time-based fitness function pushing the population to start their simulation time closer to the solution where they could discover it sooner. This experiment was replicated by Mills and Watson in 2005, in an altered form that showed canalization of genes was not necessary for a trait to become genetically assimilated [18]. By removing the '?' allele and allowing all genes a low chance to mutate, they showed that even when learning is not eliminated, the needle in the haystack may still be found.

Learning is not, however, universally beneficial and does not guarantee convergence towards genetic assimilation by itself. Mayley demonstrated that while learning does smooth out rugged landscapes generated by tunably difficult N-K equations, this can result in genetically diverse populations being rendered phenotypically indistinguishable, as all individuals who begin evaluation close enough to a local optimum can reach it through learning [19]. Mayley called this the *hiding effect*, and showed that in order to continue genetic assimilation once this point has been reached, there must be a cost to learning, either applied directly through simulated energy expenditure or indirectly through the time spent learning. Other experiments in complex environments were performed by Le *et al.*, who observed the Baldwin effect in dynamic environments, as well as among populations which could learn from each other, though the problems themselves were still simple bitstrings [20] [21]. In more complex cases, full canalization may be less desirable. In an experiment evolving neural network structure and weights, Downing observed that a percentage of synapses never canalized, and theorized that it might reflect the same tendencies biological agents had towards maintaining neural plasticity [22].

Genetic assimilation also requires a direct mapping between the genome and alterations made by learning. If the link between evolved genotype and learned phenotype is too indistinct, then mutation will be unable to replicate or assimilate learned changes, however beneficial. This is observed both in non-embodied cases [23] [24] as well as in the morphologies of embodied agents [25].

1.5.2 Embodied Instances

In 2011, Bongard *et al.* showed that including a morphological developmental trajectory in embodied agents when designing controllers for locomotion yielded significantly more robust behavior than in agents who began at the final morphological stage [26]. While both the phenotypic change and genetic assimilation of the end phenotype were externally predefined, this research laid the grounds for Kriegman *et al*'s research into the same phenomenon in soft robots [27] [28]. Their work evolved a developmental trajectory for the size of each voxel component in the robot's body, allowing for dramatic morphological change during evaluation. It was noted that the local exploration through ontogenetic change allowed for robots to stumble across high-fitness morphologies, which created points of attraction for evolution in the same manner as Hinton and Nowlan's experiments. A robot whose body ended up rolling forwards near the end of its life received a boost to its fitness that caused rolling behavior to be encouraged. The effect on fitness was so dramatic that the discovery of rolling and its assimilation are both easily-identifiable points on the final generational fitness plots.

Another instance of this phenomenon is claimed by Gupta *et al.*, who evolved morphologies alongside reinforcement-learning controllers [29]. In a variety of task environments, they developed morphologies that rapidly evolved to better facilitate learning, and observed that agents evolved in more difficult environments showed more adaptability to novel scenarios than those evolved to perform simpler tasks. All these embodied examples are significant for both the complexity of the problems compared to prior work as well as the unique ways morphology affects the development of controllers. Some morphologies appear to be more permissive to learning and exploration, while others make these tasks more difficult [28] [25].

1.6 EMBODIMENT

When we discuss embodiment, we refer to a combination of traits which define the way an agent relates to its environment. An agent is said to be *situated* if it is capable of perceiving data about its environment. It is *embodied* if it is capable of affecting its environment in some way. All of the embodied agents discussed in this research are also situated, though that is not the case universally.

Problem spaces dealing with the control and design of embodied agents has the potential to be much more complicated than non-embodied ones. This is because physical existence introduces a temporal component to all problems by default, and so when designing control we must take into account the fact that individual outputs may have long-term consequences for behavior [6]. Also worth considering is the gap between reality and simulation, where most embodied agents are developed. Since one of the most common uses of embodied agents is in robotics, which are often required to operate in difficult real-world environments, high-performing controllers capable of operating in these difficult scenarios are required. Sims [30] demonstrated the effectiveness of evolutionary algorithms in designing these controllers, and they have been a popular method for decades. However, as discussed before, the drawbacks of overly-complex problem spaces makes designing controllers for complex tasks difficult, even with the aid of evolution.

Given the desirability of high-performing embodied systems and the advantages the Baldwin Effect has been observed to confer on such systems, it becomes clear that furthering our understanding how embodiment interacts with and affects genetic assimilation can only be beneficial to the design of such systems in the future.

Chapter 2

Methods

2.1 Overview

To explore the interactions between embodiment and the Baldwin Effect, we chose to co-evolve neural network controllers and a set of Hebbian learning parameters. This approach was tested in three morphologies of varying complexity, and the resulting parameter sets and behaviors were analyzed for signs of genetic assimilation of learning. A key component of this analysis was synaptic variance, which served as a quantifiable measure of how much learning a particular individual did.

2.2 Controller Design

2.2.1 Neural Network

The controllers developed in these experiments are feed-forward neural networks with an input, hidden and output layer. The input layer has a neuron for each sensor in the robot being controlled, and the hidden layer has a number of neurons equal to the input layer. The output layer, meanwhile, has a neuron for each motor in the robot. All neurons have a *tanh* activation function, bounding their output to [-1, 1]. Each layer is fully connected to the previous layer, with synaptic weights within the range [-1, 1]. For the sake of precision, all neuron outputs and synaptic weights are rounded to the 6th decimal place during simulation. A general diagram of the neural neural networks used is depicted in Figure 2.1.

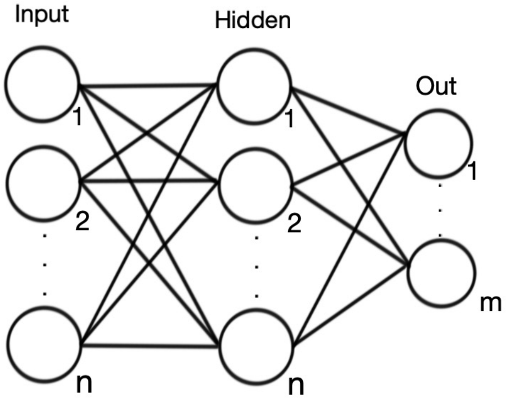

Figure 2.1: A general diagram for the neural network's layout. Each layer is fully connected to the layer previous, resulting in $N * N * M$ *synapses per network.*

2.2.2 Genome Design

Each controller has a genome which determines the weight of its synapses at simulation timestep 0 w^0_{ij}, as well as the plasticity of each synapse η_{ij}, which appears in the Hebbian update equation. Values for w^0 fall within [-1, 1], and η is bounded to [0, 1].

2.2.3 Hebbian Learning

Alongside the evolved population of neural network genomes we also develop a set of parameters for Hebbian learning, H. This parameter set, as well as the equations by which synapses are modified and the way H is evolved, is a modified version of the methods utilized by Najarro and Risi [12]. Rather than an $S \times 5$ set of parameters, each synapse s_{ij} corresponds with only four variables, $< A_{ij}, B_{ij}, C_{ij}, D_{ij}$, with control of the fifth parameter η_{ij} shifted to the genome for reasons discussed in Section 2.4.2. A, B, C, and D are all initially generated as values within [-1, 1] but are not kept within those bounds when H is updated.

After the neurons of a controller have fired during a simulation's *update* step, the potential change in weight Δw for synapse s_{ij} connecting neurons i and j with activation levels o_i and o_j at timestep t is determined by the following equation:

$$\Delta w_{ij}^{t} = \eta_{ij} * (A_{ij} * o_i^t * o_j^t + B_{ij} * o_i^t + C_{ij} * o_j^t + D_{ij})$$

Δw_{ij}^{t} is then added to w_{ij}^{t}, and the resulting value is rounded to the sixth decimal place, bounded to $[-1, 1]$, and used as the weight of s_{ij} for the network update sequence in timestep $t + 1$.

2.3 Morphologies

Three distinct morphologies were tested using these controllers. These morphologies are parametrically defined, with a common limb length shared between each of them. This was partially done to make scaling of the robots' sizes easier, and to ensure similarity between the forces the robots motors were capable of. All robots are rigid-bodied, with single-axis hinge joints providing capacity for movement and proprioceptive sensors monitoring their current rotational position. These joints' maximum deviation from their resting position is limited to $\frac{\pi}{3}$ radians. Additional sensory information is generated by "touch" sensors in robots' limbs, which reports a value of +1 if in contact with the ground and a -1 otherwise. Figures 2.2-2.4 show technical diagrams of the robots' shape and joints. While morphological complexity is a difficult concept to rigorously define, these three specific designs were chosen due to the varying size of their neural networks and number of moving parts, and are listed in order from smallest to largest.

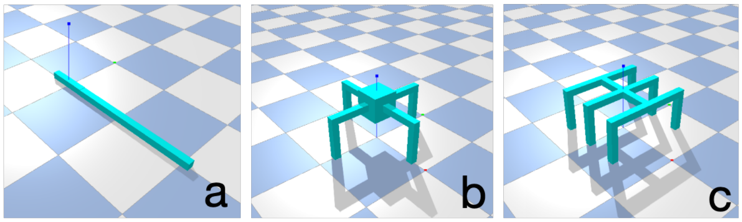

Figure 2.2: All three morphologies at rest in simulation. The current position of each of these robots defines a zero-degree rotation in their joints, from which every hinge may rotate a maximum of $\frac{\pi}{3}$ radians. Technical diagrams of the robots' bodies and joint angles may be found in Appendix A.

2.3.1 SNAKE

The first morphology, a three-jointed robot, is composed of four segments connected end-to-end. It has the smallest network, with only 11 neurons and 28 synaptic connections. The Snake has four sensors: one for each of its joints, and a touch sensor in a segment at one end.

2.3.2 Quadruped

[hbt!] The second morphology is a recreation of the quadrupedal robot used in Najarro and Risi's research, consisting of four two-jointed legs connected in a radially symmetric fashion to a central body [12]. While similar in shape, this robot has fewer degrees of freedom than the design it is based on, as its hinge joints rotate on only one axis instead of two. This robot has a touch sensor at the bottom segment of each leg and a proprioceptive sensor at each joint, for a total of 12 inputs, 8 outputs, and 240 synapses in the network.

2.3.3 Hexapod

The last morphology is a six-legged bilaterally symmetrical robot. Its legs have sensors at their lower segments similar to the Ant, though due to the increased number of legs its neural network is larger, with 18 inputs, 12 outputs, and 540 synapses. Unlike the Quadruped, the Hexapod's body-leg joints rotate along an axis perpendicular to the ground, rather than parallel.

2.4 Evolutionary Algorithms

2.4.1 Genetic Algorithm

The population P of controllers at the beginning of each generation g is composed of 30 genomes. Randomly-selected individuals are copied, and their copies are subjected to mutation. During mutation, each individual parameter has a 10% chance of being

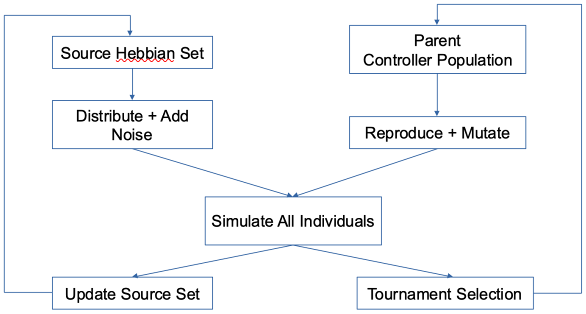

Figure 2.3: A diagram illustrating the two evolutionary algorithms running in this experiment. The co-evolution between the Hebbian parameter set and the neural network genomes means that every genome requires re-evaluation during the simulation step, to ensure that their performance is maintained when paired with the new set of Hebbian parameters.

modified. This modification consists of a new value being selected from a normal distribution centered on the parameter's current value, so that most values selected for mutation do not change drastically. If the new value falls outside the bounds set for its parameter type, it is set at the bound limit instead.

In Najarro and Risi's work, control of the plasticity parameter η falls under the mass gradient-descent evolutionary strategy responsible for refining the Hebbian parameters. We place η under the control of a genetic algorithm with point-mutation to allow for finer tuning of individual synapses, as η is the prime control parameter for how much learning a synapse will do in simulation.

A total of 30 children are produced in this manner, and the entire population - parents and children - are simulated with Hebbian parameters generated from H. Following the conclusion of simulation, each controller obtains a fitness value equal to its final horizontal displacement, regardless of direction. The highest-fitness controller in P_g is automatically saved for the population of the next generation P_{g+1}, and the remaining 29 spots are filled via randomized tournament selection. This process consists of selecting three random individuals from P_g and adding the highest-fitness among those three to P_{g+1}. After P_{g+1} has been filled, the cycle begins again in the new generation.

2.4.2 HEBBIAN EVOLUTIONARY STRATEGY

We maintain a "parent set" of Hebbian parameters H, with the properties discussed in Section 2.2.3. At the beginning of a generation, H is used to create P randomly-modified copies of itself, permuting each parameter with random noise $N(0,1) * \sigma$, where σ is an externally-defined hyper-parameter. These randomly permuted copies are each assigned to a controller in the population, and used as that controller's Hebbian rule set during simulation. After the entire population has finished simulation, H is updated by taking the sum of the deviations in each of its parameters, weighted by the fitness F of each of those copies and dampened by a learning rate α, as shown in the equation below.

$$H_{g+1} = H_g + \frac{\alpha}{P\sigma} * \sum_{p=0}^{P} F(p) * N(0,1)_p$$

Contrary to Najarro and Risi's use of this algorithm, we did not decay the values of α or σ, as imposing artificial time limits on the development of learning rules would run contrary to the aims of this research. We found that even without the imposition of decay, hebbian parameters converged to values within ±8.

2.5 SIMULATION

Robots are simulated with *Pybullet*, an open-source Python physics simulation library. The code responsible for constructing robot morphologies and managing the communication between *Pybullet* and the neural network was originally developed by Bongard [31], though extensive modifications have been made to allow for hidden neurons, reduce computation in processing neural network updates, allow for live synaptic updates, and track the neural network's state at each timestep. Robots' simulation environment consisted of a flat, featureless plane with a constant downward gravitational force of 9.8 m/s. Simulation in this environment ran for 1000 timesteps, with each timestep consisting of three steps: *sense*, *update*, and *act*. Each of these steps denotes a different part of the simulation updating, and each generates information that is used by the next stage.

In the *sense* stage, the simulation updates its physical state. Objects in mid-air are accelerated downwards, motors move, and sensors' states are updated. This stage is handled entirely by *Pybullet*'s in-built stepSimulation() function.

In the *think* stage, the controller updates its internal states. Input neurons update with the new values generated by their associated sensors, and pass these values forward through the network. The output neurons generate new angles for the motors to take on in the next stage. After every neuron's new value has been calculated, synaptic weights are updated according to the equations discussed in Section 2.3. This is also the stage at which all synaptic weight and neuron activation is recorded for post-simulation analysis.

Lastly, in the *act* stage, the robot's morphology acts on the information processed by the controller. Each hinge joint's target angle is set to a percentage of its maximum rotational range equal to the activation of its associated motor neuron. An output of 0 causes a joint to move towards its resting position, a negative number indicates contraction, and a positive number extension. These target angles translate into physical acceleration of the connected body parts in the *sense* stage when the loop repeats.

Following the conclusion of 1000 simulation loops, the final horizontal displacement of the robot and its synaptic activity are recorded, passed back to the main evolutionary loop, and utilized as that controller's fitness.

Chapter 3

Results

3.0.1 Quantitative Analysis of Phenotypic Change

An obstacle encountered when attempting to measure our findings comes in the way that the Baldwin Effect is typically observed. As is noted in Chapter 1, prior experiments investigating the Baldwin Effect in action utilized problem spaces where an optimal outcome was known and the problem space was discrete, or at least continuous over a small number of dimensions. This resulted in solution sets whose approach towards an optimum could be easily observed and measured, and whose canalisation of the optimal phenotype was similarly straightforward.

A highly non-linear problem space, such as is presented here, has no known optimal solution. Observations of the Baldwin Effect in embodied agents in the past have often been post-hoc, with examples drawn from particularly dramatic outcomes where the gradual canalisation of a learned strategy can be easily observed qualitatively [28]. Since we modify only the controllers of our robots, not the morphologies, we cannot expect to be able to easily observe similar results visually, and thus the problem emerges of finding a way to quantify the genetic assimilation of learning.

The Baldwin effect requires two events to occur - first, learning must discover a desirable phenotype, and second, evolution must then push the genome towards performing that phenotype as an inherent trait. Thus, the amount of change an individual undergoes over the course of its lifetime can be said to be an approximate measure of how much (or how little) it has assimilated the behavior it learns. An individual who performs less learning than its predecessors but maintains the same fitness (or does even better) can be said to be more "baldwinized". It is important to note that we cannot use the shortcut of looking directly at the genome and measuring the number of non-learning traits, as Hinton and Nowlan did, to determine whether behavior is fully assimilated. As Mayley showed, canalisation is not equivalent to genetic assimilation [19]. Additionally, the nature of the neural network controller may *require* some synapses to remain fully plastic in order to maintain a high-fitness behavior [22].

As the development of our robots centers around the modification of their synaptic weights, we observe the lifetime variance of those weights to determine the amount of development the robots' neural networks undergo. The higher synaptic variance is across the network, the more synapses are being modified, and the more learning is being performed. We measure the learning an individual network p has done as the mean of the variance of synaptic weights:

$$V_p = \frac{1}{S} \sum_{I,J} var(w_{ij})$$

To show that this reduction in learning is due to genetic assimilation, we track the population's fitness values. The performance (and thus, the fitness) of a robot is the result of two factors: its starting conditions (genome) and its interactions with the environment (learning). If learning is significantly reduced while fitness remains the same, then the genome must be taking more responsibility for maintaining performance. Thus, a population that has "baldwinized" its learned behavior will at least maintain its fitness while significantly reducing learning, thus indicating a transfer of behavior to inherent traits.

$$F_p^0 <= F_p^{199}, V_p^0 >> V_p^{199}$$

Analysis of generational fitness and variances of 50 separate evolutionary runs for controllers of each morphology shows controllers for all three morphologies managed to significantly increase their fitness and maintain that increase while significantly decrease synaptic variance in their networks by generation 200.

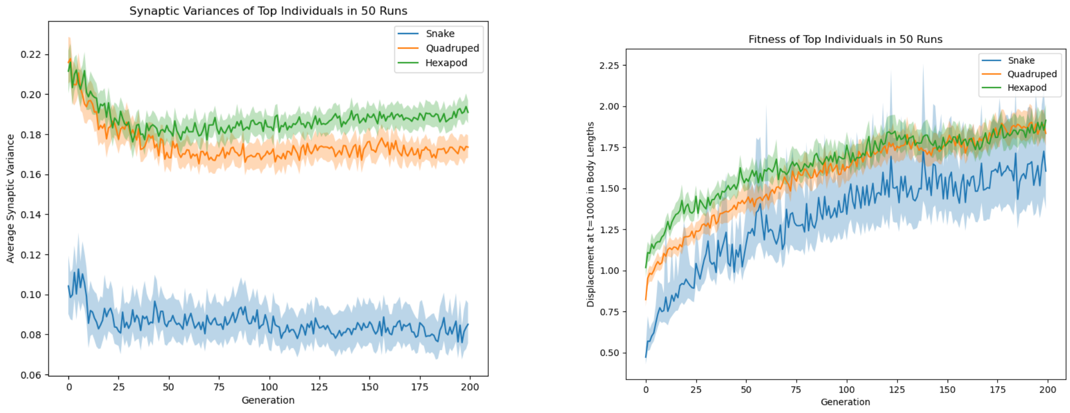

Figure 3.1: Plots of average network variance (left) and final displacement (right) of the best-performing individuals of each morphology across 50 200-generation evolutionary trials. Independent T-tests find all three morphologies showed significant decreases in variance ($p<.05$*) and increases in fitness (*$p\ll.005$*), indicating they have genetically assimilated some portion of their behavior.*

3.0.2 BEHAVIOR OF EVOLVED LEARNING PARAMETERS

Having established that morphology does not appear to affect the ability of the Baldwin effect to take effect, we investigate further to examine the ways in which behavior became genetically assimilated. The governing equation for synaptic change, and thus, learning, contains two main components capable of reducing Δw: η_s, the plasticity of the synapse, and $(ABCD)_s$, the Hebbian parameters operating on that synapse. Should either η or all four Hebbian parameters be reduced to zero, then Δw would also be zero, and the synapse would not change.

Analysis of η values over evolutionary time found that the number of synapses with a plasticity of 0 increased rapidly over the first 30 generations before asymptotically approaching about 33 percent of the network's synapses. This pattern and proportion held true for all three morphologies. At the same rate, and with roughly the same asymptote, the number of synapses with their plasticity set to the maximum value of 1 also increased. Hebbian parameters, on the other hand, failed to reach their zero-learning case in any synapse across all morphologies in any run.

Further comparison of non-learning synapses with their attached learning parameters revealed a third case for synaptic non-learning. Due to the constraints limiting synaptic weight to within [-1, 1], synapses which existed at either boundary were able to spend the entire simulation pushing up against it. Their plasticities and Hebbian parameters were all non-zero, but due to the network's behavior they were constantly attempting to reach more extreme values and being reset. Since synaptic weights were bounded before being recorded at each timestep, this registered as a net change of 0, and thus the synapses did not learn. Analysis of the proportions of "bounded"

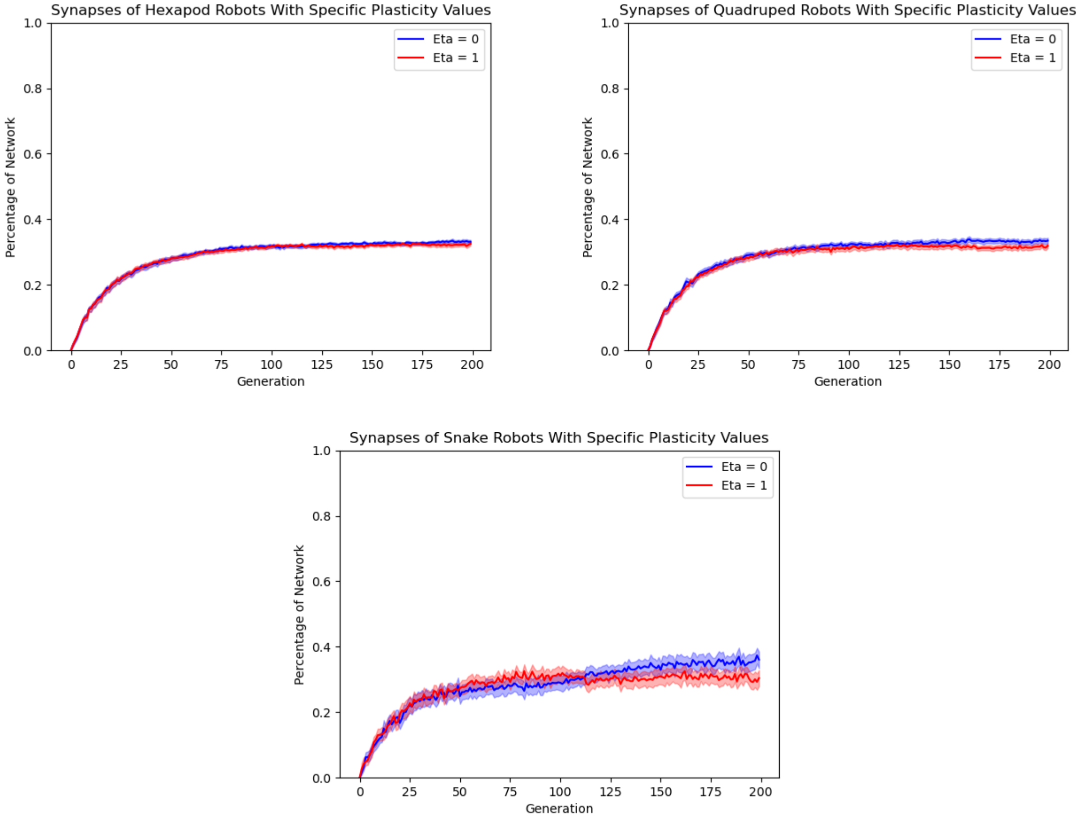

Figure 3.2: Shown are generational development of synaptic plasticities for each morphology. An independent T-test with Bonferroni correction finds the final proportion of synapses with zero plasticity in the network is not different enough in any of the three morphologies to be significant ($p > .025$). The plasticity of these synapses appears to be polarizing, with the number of synapses with the maximum η of 1 increasing at more or less the same pace as those with a value of 0.

synapses in networks showed a statistically significant difference between morphologies. Contrary to the generational development of eta, however, the percentage of "bounded" synapses in each morphology's network *decreased* over evolutionary time. This is likely due to these synapses having their η values reduced to 0 transferring them from one "non-learning" category to the other.

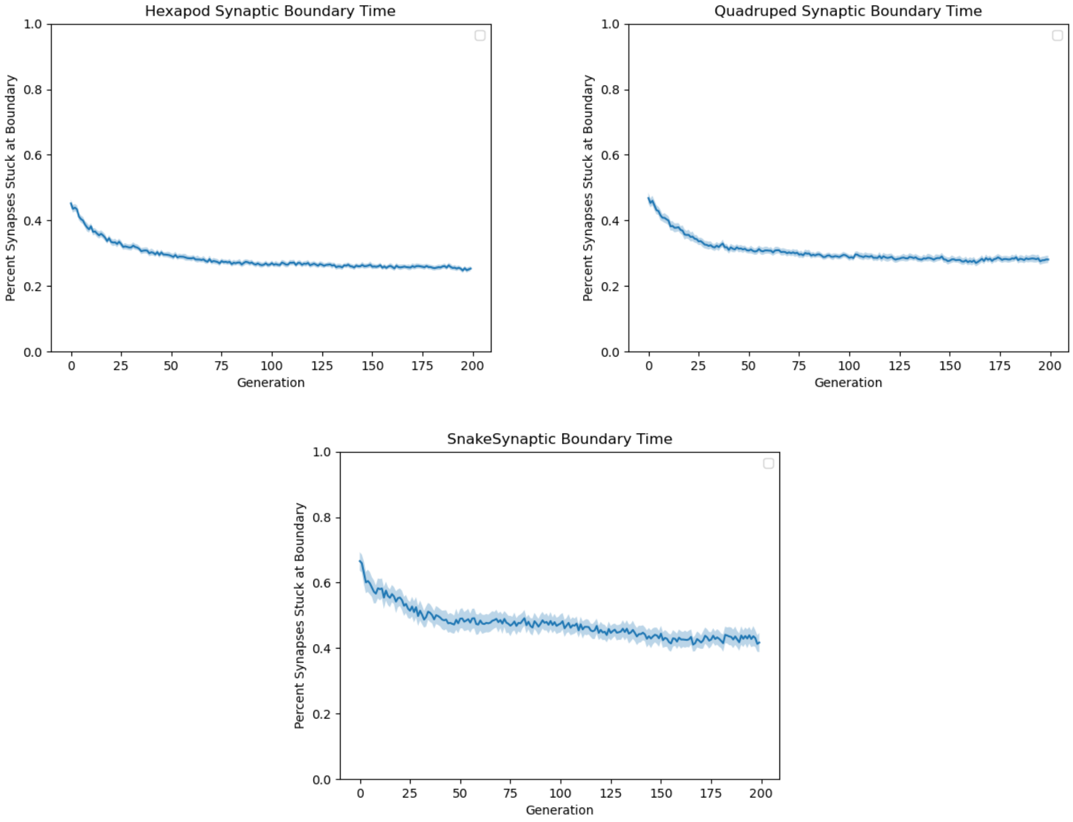

Figure 3.3: Shown are plots of the proportions of each network's synapses which are "bound" at their maximum (or minimum) values due to the pressures placed on them by their learning rules over evolutionary time. Controllers for all three morphologies saw a significant decrease in this proportion between generations 0 and 200, though the final percentages were significantly different.

CHAPTER 4

DISCUSSION AND FUTURE WORK

In this work we have introduced a method for measuring the degree to which a neural network has genetically assimilated behavior learned to solve a complex task where the optimal solution is not known, and used it to study the Baldwin effect in three morphologically-distinct embodied control problems. Our results suggest that while genetic assimilation of behavior will occur regardless of morphology - all three controller types successfully completed some amount of genetic assimilation in the generational time provided - the amount of assimilation depends on morphology. Additionally, while morphology has an effect on genetic assimilation through learned behavior, the rate and capacity for canalization appears to be constant regardless of morphology.

It is unclear why canalization was similar across networks of different complexity that were operating different control problems, and whether the fraction of synapses locked in this manner is significant. One explanation could be that the lack of further canalization is an instance of Mayley's "hiding" effect. Hebbian plasticity is a powerful learning tool capable of rapid change, and if controllers are traversing

Morphology	$\eta = \mathbf{0}$	**bounded**	**total** %
Snake	.361	.417*	.778*
Quadruped	.335	.281*	.616*
Hexapod	.331	.253*	.585*

Table 4.1: Causes for zero-variance in synapses, displayed as the mean percentage of the network's synapses they represent. Asterisks indicate that values have been determined to be significantly different from other values in the same column by an independent T-test (p«.001).

phenotype-space *too* quickly, then the effects of starting at a slightly poorer location in the fitness landscape might not make themselves as apparent. This would explain the lack of further canalization past a certain point, but not why all three morphologies, which represent presumably entirely different fitness landscapes, would cease canalizing at almost exactly the same point. A confounding factor in identifying patterns in the canalization process is that comparison between individuals within a given morphology type is made difficult by the networks' fully-connected hidden layer. Because hidden neurons are topologically interchangeable, the same synapse may serve a different function between two neural networks with identical structures. This means that even if certain structures were being converged upon within the 50 runs completed, a much deeper analysis would be required to identify such patterns.

The large number of synapses locked at their extreme values is an expected result - Najarro and Risi observed the tendency towards extremes in their findings [12] and Miller noted that Hebbian rules which followed a subtractive bias such as the one present in the *ABCD* Hebbian strategy tended towards extremes, "sharpening" synapses which played a major role in behavior while disregarding others. This behavior may function as a sort of "holding pattern" for the neural network - synapses are held more or less in stasis by their boundaries, but remain ready to respond to any changes in input patterns, similar to the ways biological systems retain plasticity in otherwise stable phenotypes, ready to respond to perturbations in their environments. This would be easily tested with the introduction of novel obstacles in the agents' environment. If encountering new stimuli causes previously-stable "bounded" synapses to assume new configurations, then we can deduce that their prior state was specific to the old environment, and that their continued high plasticity is necessary for the adaptability of the controller.

It is well-established that morphology and control are deeply linked [32], and prior literature's investigation into morphological development has shown that certain morphologies are significantly more permissive to evolve controllers for [28]. The same has been found to be true for neural network topologies, to the extent that a properly-evolved neural network structure may perform complex tasks with no tuning at all [33]. The differing levels of "sharpening" created by learning could be a result of the Hebbian learning rules attempting to simulate an alteration to the network's topology by maximizing the most-used connections as much as possible. Synaptic connections with higher weight carry more importance, and can potentially drown out the signal from neurons with lower-fidelity connections. If it is the case that phenotypic change is attempting to simulate an alteration to neural network topology, and the alterations made depend on the morphology being controlled, then this approach may be useful for optimizing neural network topologies to specific embodied agents.

BIBLIOGRAPHY

[1] Hugo de Garis. Genetic programming: Building artificial nervous systems using genetically programmed neural network modules. *Proceedings of the 7th International Conference on Machine Learning*, pages 132–139, 1990.

[2] Marvin Minsky. Steps toward artificial intelligence. *Proceedings of the IRE*, 49:8–30, 1961.

[3] D. E. Goldberg. *Genetic Algorithms in Search, Optimization, and Machine Learning.* Addison-Wesley, 1989.

[4] Michael D. Schmidt and Hod Lipson. Age-fitness pareto optimization. In *GECCO '10: Proceedings of the 12th annual conference on Genetic and evolutionary computation*, pages 534–544. ACM, 2010.

[5] Joel Lehman and Kenneth O. Stanley. Abandoning objectives: Evolution through the search for novelty alone. *Evolutionary Computation*, 19:189–223, 2011.

[6] Andrea Soltoggio, Kenneth O. Stanley, and Sebastian Risi. Born to learn: the inspiration, progress, and future of evolved plastic artificial neural networks. *Neural Networks*, 108:48–67, 2018.

[7] Donald Hebb. *The Organization of Behavior.* Wiley & Sons, 1949.

[8] David Willshaw and Peter Dayan. Optimal plasticity from matrix memories: What goes up must come down. *Neural Computation*, 2:85–93, 1990.

[9] Kenneth D. Miller and David J. C. Mackay. The role of constraints in hebbian learning. *Neural Computation*, 6:100–126, 1994.

[10] Dario Floreano and Francesco Mondada. Evolution of plastic neurocontrollers for situated agents. *From Animals to Animats 4: Proceedings of the Fourth International Conference on Simulation of Adaptive Behavior (SAB 1996)*, pages 402–410, 1996.

[11] Tim Salimans, Jonathan Ho, Xi Chen, and Ilya Sutskever. Evolution strategies as a scalable alternative to reinforcement learning. *ArXiv*, abs/1703.03864, 2017.

[12] Elias Najarro and Sebastian Risi. Meta-learning through hebbian plasticity in random networks. In *Advances in Neural Information Processing Systems*, volume 33, pages 20719–20731. Curran Associates, Inc., 2020.

[13] Andrea Ferigo, Giovanni Iacca, Eric Medvet, and Federico Pigozzi. Evolving hebbian learning rules in voxel-based soft robots. *IEEE Transactions on Cognitive and Developmental Systems*, 2022.

[14] J. Mark Baldwin. A new factor in evolution. *The American Naturalist*, 30:441–451, 1896.

[15] C. H. Waddington. Canalization of development and the inheritance of acquired characters. *Nature*, 150:563–565, 1942.

[16] Scott F. Gilbert. Diachronic biology meets evo-devo: C.h. waddington's approach to evolutionary developmental biology. *American Zoologist*, 40:729–737, 2000.

[17] Geoffrey E. Hinton and Steven J. Nowlan. How learning can guide evolution. *Complex Systems*, 1, 1996.

[18] Rob Mills and Richard A. Watson. *Genetic Assimilation and Canalisation in the Baldwin Effect*, pages 353–362. Springer Berlin Heidelberg, 2005.

[19] Giles Mayley. Landscapes, learning costs, and genetic assimilation. *Evolutionary Computation*, 4:213–234, 1996.

[20] Nam Le, Anthony Brabazon, and Michael O'Neill. *How the 'Baldwin Effect' Can Guide Evolution in Dynamic Environments*, pages 164–175. Springer International Publishing, 2018.

[21] Nam Le, Michael O'Neill, and Anthony Brabazon. Evolutionary consequences of learning strategies in a dynamic rugged landscape. In *Proceedings of the Genetic and Evolutionary Computation Conference*, pages 812–819. ACM, 2019.

[22] Keith L Downing. The baldwin effect in developing neural networks. In *Genetic and Evolutionary Computation*, pages 555–562. ACM, 2010.

[23] Kim W. C. Ku and M. W. Mak. *Empirical Analysis of the Factors That Affect the Baldwin Effect*, pages 481–490. Springer Berlin Heidelberg, 1998.

[24] Kim W.C. Ku. Enhance the baldwin effect by strengthening the correlation between genetic operators and learning methods. In *IEEE International Conference on Evolutionary Computation*, pages 3302–3308. IEEE, 2006.

[25] Federico Pigozzi, Federico Julian Camerota VerdĂš, and Eric Medvet. How the morphology encoding influences the learning ability in body-brain co-optimization. In *Genetic and Evolutionary Computation Conference (GECCO '23)*. ACM, 2023.

[26] Josh Bongard. Morphological change in machines accelerates the evolution of robust behavior. *Proceedings of the National Academy of Sciences*, 108:1234–1239, 2011.

[27] Sam Kriegman, Nick Cheney, Francesco Corucci, and Josh C. Bongard. A minimal developmental model can increase evolvability in soft robots. In *Proceedings of the Genetic and Evolutionary Computation Conference*, pages 131–138. ACM, 2017.

[28] Sam Kriegman, Nick Cheney, and Josh Bongard. How morphological development can guide evolution. *Scientific Reports*, 8, 2018.

[29] Agrim Gupta, Silvio Savarese, Surya Ganguli, and Li Fei-Fei. Embodied intelligence via learning and evolution. *Nature Communications*, 12, 2021.

[30] Karl Sims. Evolving virtual creatures. In *Proceedings of the 21st annual conference on Computer graphics and interactive techniques*, pages 15–22. ACM, 1994.

[31] Josh Bongard. Pyrosim, 2022. `https://github.com/jbongard/pyrosim`.

[32] Rolf Pfeifer and Josh Bongard. *How the body shapes the way we think: a new view of intelligence*. MIT press, 2006.

[33] Adam Gaier and David Ha. Weight agnostic neural networks. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc, 2019.

Appendix A

Technical Diagrams of Robot Morphology

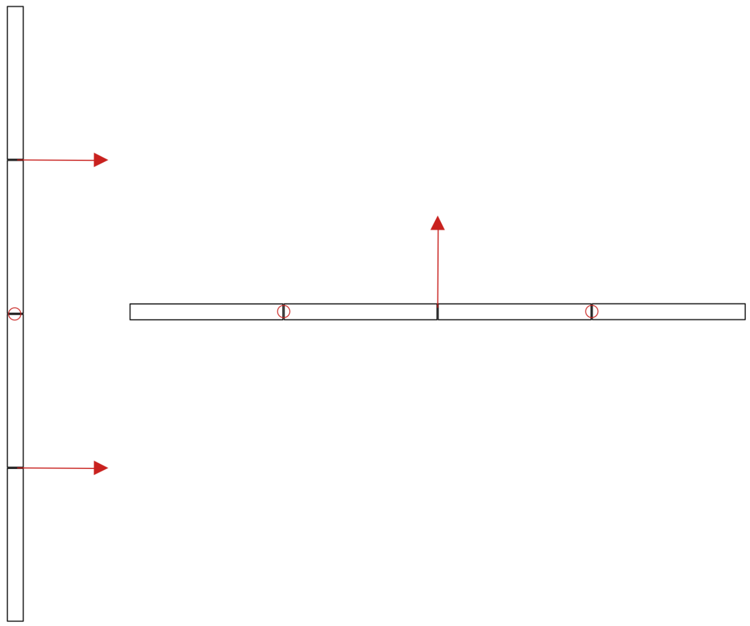

Figure A.1: A diagram of the Snake, with views from the top (left) and side (right). Joint axes of rotation are indicated by red arrows.

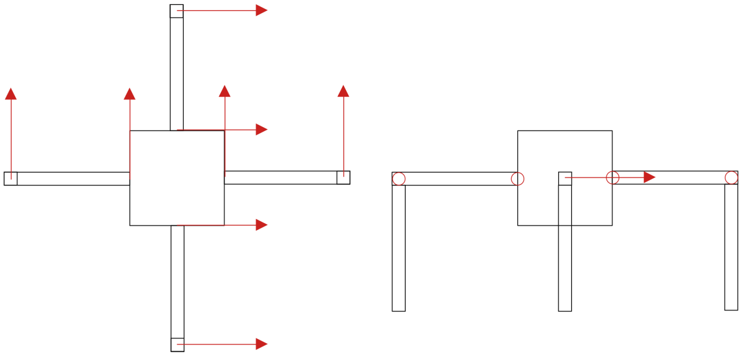

Figure A.2: A diagram of the Quadruped, with views from the top (left) and side (right). Joint axes are indicated by red arrows.

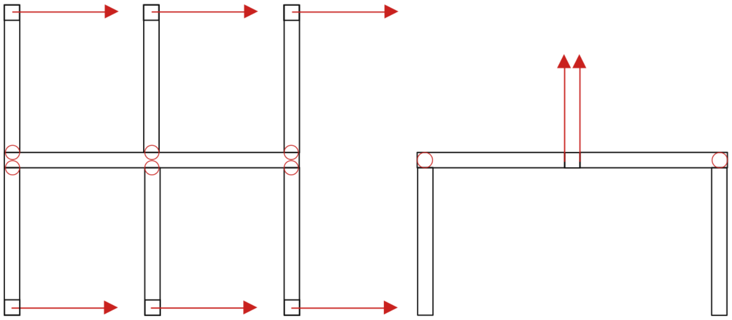

Figure A.3: A diagram of the Hexapod, with views from the top (left) and front (right). Joint axes are indicated by red arrows. Note that, unlike the Quadruped, the hip joints are normal to the ground rather than parallel.