Spontaneous Evolution of Modularity in Neural Networks for Robot Locomotion

Christian Schluchter

Submitted for the degree of MSc ETH in Robotics, Systems and Control

Supervisors:
Prof. Dr. Josh C. Bongard
Morphology, Evolution & Cognition Lab, University of Vermont, USA

Prof. Dr. Fumiya Iida
Bio-Inspired Robotics Lab, ETH Zurich, Switzerland

August 2011
Abstract

Modular neural networks have a number of advantages when used to control robots. They reduce the number of connections and thus the computational power required to update them compared to fully connected networks. Various literature also suggests that modularity increases robustness and evolvability of neural networks. Hence, modularity has the potential to pave the way towards adaptive and resilient controllers for complex robots that previously proved computationally infeasible.

To explore these issues, this work shows the evolution of modularity in neural networks, modelled as dynamically autonomous random boolean networks, that control robots to exhibit phototaxis in a simulated environment. In experiments on a wheeled robot networks that settle into fixed point attractors were studied, whereas in experiments on a legged robot networks that settle into cyclic attractors were considered. In further experiments it was investigated how the results scale to larger networks and robots were excluded: desired network dynamics were selected for instead of selecting based on their influence on robot behavior. It was found that modularity emerged when selecting for cyclic attractors in one part of the network and selecting for fixed point attractors in the other part.
Declaration

I hereby declare that this thesis is my own work and effort and that it has not been submitted, either in the same or different form, to this or any other university for a degree.

Signature:
Acknowledgements

It is a pleasure to thank Josh Bongard for the generous offer of time and experience as well as for the contagious commitment and motivation. Many thanks also to Fumiya Iida, who allowed for this thesis to come about.

Furthermore, I thank my parents for the support and for encouraging my curiosity about how the world works; without that this thesis would not exist.

The computational resources provided by the Vermont Advanced Computing Center, which is supported by NASA (NNX-08AO96G), are gratefully acknowledged.

Christian Schluchter, schluchc@ee.ethz.ch
Contents

1 Introduction

2 Background and Related Work
   2.1 Predefined Modular Structure . . . . . . . . . . . . . . . . . . . . . . . . . . . . 3
   2.2 Modularity as a Result of High-Level Processes . . . . . . . . . . . . . . . 4
   2.3 Spontaneous Evolution of Modularity . . . . . . . . . . . . . . . . . . . . . . 4

3 Methods
   3.1 The Task . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 6
   3.2 The Robot Bodies . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 6
   3.3 The Controllers . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 7
   3.4 The Evolutionary Algorithm . . . . . . . . . . . . . . . . . . . . . . . . . . . . 9
   3.5 Modularity . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 10

4 Results
   4.1 Wheeled Robot . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 13
   4.2 Legged Robot . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 15
   4.3 Explicit Selection of Cyclic Activity Patterns . . . . . . . . . . . . . . . . . . 18
   4.4 Benchmark . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 24

5 Discussion and Analysis
   5.1 Selection for Fixed Point Attractors . . . . . . . . . . . . . . . . . . . . . . . . 27
   5.2 Selection for Cyclic Attractors . . . . . . . . . . . . . . . . . . . . . . . . . . . . 28
   5.3 Hidden Nodes Affiliation . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 30
   5.4 Why Modularity Arises . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 30
   5.5 Scaling the Networks . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 32
   5.6 Future Work . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 33

6 Conclusion

iv
# List of Figures

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.1</td>
<td>The Task Environments</td>
<td>7</td>
</tr>
<tr>
<td>3.2</td>
<td>The Robot Bodies</td>
<td>8</td>
</tr>
<tr>
<td>4.1</td>
<td>Successful Controllers for the Wheeled Robot</td>
<td>14</td>
</tr>
<tr>
<td>4.2</td>
<td>Simulation Results of a Successful Controller for the Wheeled Robot</td>
<td>15</td>
</tr>
<tr>
<td>4.3</td>
<td>Successful Runs and Modularity in the Experiments on the Robots</td>
<td>16</td>
</tr>
<tr>
<td>4.4</td>
<td>Example of a Successful Controller for the Legged Robot</td>
<td>17</td>
</tr>
<tr>
<td>4.5</td>
<td>Simulation Results of a Successful Controller for the Legged Robot</td>
<td>18</td>
</tr>
<tr>
<td>4.6</td>
<td>Average Connectivity of Successful Controllers for the Legged Robot</td>
<td>19</td>
</tr>
<tr>
<td>4.7</td>
<td>Desired Angles when Selecting for Varying Frequency or Amplitude</td>
<td>21</td>
</tr>
<tr>
<td>4.8</td>
<td>Average Connectivity of Controllers when Selecting for Varying Amplitude</td>
<td>22</td>
</tr>
<tr>
<td>4.9</td>
<td>Successful Runs and Modularity when Selecting for Varying Amplitude</td>
<td>23</td>
</tr>
<tr>
<td>4.10</td>
<td>Average Connectivity of Controllers when Selecting for Varying Frequency</td>
<td>24</td>
</tr>
<tr>
<td>4.11</td>
<td>Successful Runs and Modularity when Selecting for Varying Frequency</td>
<td>25</td>
</tr>
<tr>
<td>4.12</td>
<td>Modularity in Random Walks</td>
<td>26</td>
</tr>
<tr>
<td>5.1</td>
<td>Comparison of Speed Variation by Amplitude, Frequency or Both</td>
<td>29</td>
</tr>
<tr>
<td>5.2</td>
<td>Module Affiliation of the Hidden Nodes for the Legged Robot</td>
<td>31</td>
</tr>
<tr>
<td>5.3</td>
<td>Module Affiliation of the Hidden Nodes when Selecting for Dynamics</td>
<td>32</td>
</tr>
</tbody>
</table>
List of Tables

3.1 Initial and Final Node Activities for the Wheeled Robot . . . . . . . . . . 8
3.2 Node Activities and Desired Angles for the Legged Robot . . . . . . . . . 9
4.1 Overview of the Experiments . . . . . . . . . . . . . . . . . . . . . . . . . 12
5.1 Alternative Initial and Final Node Activities for the Wheeled Robot . . . 28
1 Introduction

Traditional control approaches have not yet proven successful in tasks that require robots to autonomously adapt to unforeseen changes in their environments, their morphologies or their tasks. Such abilities are crucial to robots belonging to the category of autonomous and adaptive machines rather than elaborate mechanical tools. The majority of the robots that are deployed today, such as industrial robots, consequently exclusively operate in the well-known, static and structured environments they are designed to work in, such as car production lines.

A promising approach to generating more adaptive and resilient robots than traditional control approaches have yielded so far is provided by the relatively new field of evolutionary robotics introduced in 1993 by Cliff et al. [4, 20]. Evolutionary robotics is inspired by the Darwinian principle and makes use of evolutionary algorithms to optimize neural networks that control robots. One major difficulty in evolutionary robotics is the scaling up of the successes achieved in basic tasks, such as locomotion or taxis, to more complex problems. Everyday tasks, as done by humans with little conscious effort, require a great amount of adaptivity, spontaneity and resilience. To be successful in such complex tasks, robots need to be able to deal with uncertainties, changes, and previously unencountered events.

Robots that perform complex tasks require a great number of sensors and actuators. Hence, neural networks controlling such robots are comprised of a great number of nodes and connections. In a fully connected neural network, the number of connections grows quadratically with the number of nodes. Therefore, fully connected neural networks become too computationally expensive or even infeasible for controlling complex robots. One way around this complexity barrier is to introduce modularity and/or component reuse. In a modular network, the nodes within its modules are densely interconnected, while there is only sparse connectivity between nodes belonging to different modules. However, the manual design of neural networks is not intuitive and highly limited by designer’s bias. Due to this limitation, manually-designed modularity often is constrained to manually-designed functional modularity. Functional modularity denotes a partitioning of the network into modules each carrying out a different function, e.g. a controller is partitioned into modules, each controlling one leg of a robot for the task of quadrupedal locomotion [26]. If evolution takes over the partitioning into modules, any form of structural modularity that is beneficial to the assigned task can arise. Structural modularity denotes the partitioning of the network into groups of nodes that feature many connections within the groups, but only few connections between the groups.

Aside from a possible solution to the scalability problem in robotics, modularity is a fundamental aspect of biological organization and has been found to have positive properties in both the biological and artificial neural networks literatures. Most prominently,
modular networks are supposed to exhibit higher evolvability and robustness compared to non-modular networks [7, 15, 21, 30]. Due to the sparse connectivity between modules, structural modularity allows the improvement of some modules without interfering with other modules. This enables a network to improve in some subgoals without losing performance in other subgoals.

Wagner [30] suggested that modularity can spontaneously arise in biological organisms as a result of directional selection acting on one part of the organisms’ phenotype and stabilizing selection acting on the other part. Lipson et al. [28, 16] later showed that modularly varying subgoals lead to the spontaneous evolution of modularity. Several publications based on these findings showed the spontaneous evolution of modularity under different conditions [12, 8, 3].

This thesis aspires to continue this stream of work and implements these ideas to show the spontaneous evolution of modularity in neural networks that control driving behavior in wheeled robots and locomotion in legged robots. This was accomplished by evolving robots to exhibit phototaxis: they must move toward a light source placed at different positions in four environments. The light source placements were chosen such that two environments always share a subtask. To succeed the robot must drive or walk toward the light source which is placed in one of four possible locations: either front left or front right of the robot and close or far from it. On a simulated wheeled robot this is accomplished using a neural network controller that falls into a fixed point attractor such that constant torque is applied to the wheels. On a simulated legged robot it is accomplished using a controller that falls into a periodic attractor in which oscillations in torque are applied to the legs. For this reason, experiments on the wheeled robot allow for the investigation of the evolution of modularity when selecting for fixed point attractors, and experiments on the legged robot allow for the investigation of the evolution of modularity when selecting for cyclic attractors. In both experiments, increases in modularity in the neural networks led to increases in performance.

In experiments where selection was applied directly to larger networks’ dynamics instead of how they influence the behavior of a robot, the spontaneous evolution of modularity was achieved by selection for cyclic attractors that either vary oscillation frequency or amplitude depending on the initial conditions. These experiments provide an insight into how the results achieved in the small networks that control the robots might scale with a growing number of nodes.

The next section provides an overview of related work. The methods used in this work are presented in section 3, followed by the results. The thesis concludes with a discussion of the results and a final statement.
2 Background and Related Work

The literature about modularity in artificial neural networks can roughly be divided into three groups. Chronologically, papers first appeared that describe the mainly manual implementation of modular networks where evolution was only employed to solve sub-problems such as how to combine predefined modules or how to optimize the weights in predefined modules. Thus modularity, as a measure comparing the number of intra-module connections to inter-module connections, does not emerge during an evolutionary run, but rather must be manually built into the network. The literature reporting this kind of modularity is summarized in subsection 2.1. Another substantial group of papers describes the evolution of modularity as a result of high-level network transformations applied by evolutionary algorithms. An overview of this group is presented in subsection 2.2. The third group, of which this work is a part, describes the spontaneous emergence of modularity as a result of particular conditions that favor modular networks, but that neither use any high-level processes to support the development of modularity, nor explicitly reward for when it appears. This part of the literature is described in the last subsection.

2.1 Predefined Modular Structure

Pfeifer and Scheier [22] developed robot controllers by specifically designing modules and then letting evolution figure out how to combine them. Nolfi [19] gave more control to evolution by assigning the task to design the modules themselves to evolution. This was done on different architectures including a modular architecture with two predefined modules and an architecture referred to as an ‘emergent modular architecture’. In this architecture the partition into modules was not done by the designer himself, however, the number of available neural modules, the architecture of each module and the mechanisms that determine their interaction was predefined. Nolfi compared the performance of the different network architectures and showed that architectures that allow for modularity build networks that perform better as controllers for robots.

Nolfi also investigated the evolved modules and found that structural modules do not necessarily match to motion primitives observed by an observer and concluded that this imposes a major limitation to approaches where modules are manually built. A reason is that it is difficult for the designer to know in advance what module partition performs best. This is an important argument in favor of the spontaneous evolution of modularity, as evolution can partition the network into modules most beneficial to the robot without being constrained a priori as to how the modules should be designed.

Contrary to the above mentioned approaches, in this work no modular structure is pre-built into the network. Each node can be connected to each other node and all nodes
have the same properties. Where a connection occurs and if it is inhibitory or excitatory is solely determined by mutation within the evolutionary algorithm.

2.2 Modularity as a Result of High-Level Processes

Angeline and Pollack [2] used a specific mutation operator to promote the emergence of modularity. This operator minimized the portion of the network subject to evolution by 'freezing' certain parts of the networks while further evolving others.

Hornby et al. [11] constructed modular neural networks by use of a generative design grammar subjected to an evolutionary process. This process allowed the creation of simple modules out of basic elements that in turn could then be used to build more complex modules out of the simple modules.

Valsalam and Miikkulainen [27] presented an algorithm that identified symmetries in the robot body and built modular neural networks to control multi-legged robots. A fixed architecture for the modules was used, and evolution optimized the weights of the connections between the modules as well as the weights and biases of the connections and nodes, respectively, within the modules.

These studies all led to the evolution of modularity in neural networks by use of high-level processes that actively generate modularity. High-level network operations have a bias built in that may produce modular networks even if the task does not call for it, e.g. copying a densely connected subnetwork multiple times is likely to produce a modular network. In contrast to that approach, this work focuses on the spontaneous evolution of modularity in a neural network exclusively as a result of the beneficial performance of modular networks in the chosen task. The mutation operator employed biases networks toward sparsity, but not modularity. By only allowing change to individual connections, any modularity in evolved networks thus must provide some advantage over non-modular networks for the task at hand.

2.3 Spontaneous Evolution of Modularity

Kashtan and Alon [12] showed the spontaneous evolution of modularity by repeatedly switching between several goals, each made of a different combination of subgoals during artificial evolution. They showed the evolution of modularity in electrical circuits selected to perform different boolean functions, each made up of a different combination of boolean operators. They also demonstrated the evolution of modularity in neural networks selected to perform pattern recognition. They implemented a binary valued 4-pixel-wide by 2-pixel-high retina divided into a left and a right section. The problem consisted of recognizing a combination of one pattern on the right side and the same set of patterns on the left side. The overall problem of recognizing the combination of both patterns therefore consisted of modular subproblems. It was later referred to as the Retina Problem. Interestingly, when HyperNEAT [24], a popular algorithm to evolve neural networks, was applied to the Retina Problem by Clune et al. [6] it performed poorly and failed to produce modular solutions. An extension to HyperNEAT called
HyperNEAT-LEO, recently introduced by Verbancsics and Stanley [29], then produced modular solutions by allowing heterogeneous connectivity based on the distance of the interconnected nodes.

It was suggested by Lipson et al. [28, 16] that modularly varying subgoals lead to the spontaneous evolution of modularity. It seems that Lipson’s hypothesis is closely related to Wagner’s [30] earlier hypothesis that modularity can spontaneously arise as a result of applying directional selection to one part of the phenotype and stabilizing selection to the other part. Modularity is supposed to be beneficial under these circumstances, as it minimizes the number of inter-module connections that might impede the stabilization of the module under stabilizing selection.

Espinosa-Soto and Wagner’s implementation [8] of this principle succeeded in the spontaneous evolution of modularity in autonomous boolean networks that modelled genetic regulatory networks. They first selected for networks that attain a fixed-point attractor and upon success further generations were evolved such that half of the nodes in the network attain that same fixed point attractor while the other half fall into a different fixed point attractor based on different initial conditions. Importantly, half of the node activities were identical in the two sets of desired boolean final conditions and half were converse. Modularity increased when selection for networks that attain both fixed point attractors was applied and further increased when a third fixed-point attractor was introduced and selection for networks that attain all three attractors was applied.

Bongard [3] applied a similar idea to an autonomous random boolean network that controls a robot gripper. The task consisted of the gripping of an object which could be in two different positions and have two different sizes. The different object positions only influenced the behavior of the robot’s arm, which brings the gripper to the object position, while the size of the object only influenced the required behavior of the gripper, which grasps the object. This instance of modularly varying subgoals contains the same problem of attaining different fixed point attractors with partially identical and partially converse node activities as in Espinosa-Soto and Wagner’s paper. Consequently, modularity evolved separating the network into one module controlling the motion of the robot’s arm and one module controlling the motion of the gripper.

In this work, the task is translated to fit into the framework of robot locomotion. This was accomplished by selecting for the ability to approach a light source using a robot in which one part of the body steers and the other part of the body supplies propulsion. The problem was first tackled on a wheeled robot, where speed and steering are controlled by the network and thus fixed-point attractors are sufficient to reach the light source. Then the experiment was extended to a legged robot so that the controlling networks must fall into cyclic attractors, as oscillatory leg movement is needed to displace the robot.
3 Methods

3.1 The Task

The design of the task and the robot’s morphology are crucial to the successful evolution of modular neural controllers. The principle of modularly varying subgoals discussed above or the principle of applying stabilizing selection to one part of the phenotype and directional selection to the other part are here incorporated into a task environment that requires phototaxis performed on a flat ground plane in a simulated environment. To simulate the robots Open Dynamics Engine (ODE)\(^1\) was used to provide a physically realistic environment. The light source that the robot must approach can be in one of the four different locations: left close (LC), right close (RC), left far (LF) or right far (RF) relative to the robot’s starting position, as shown in Fig. 3.1 and the robot must walk or drive to the light source. The locations were chosen such that the robot’s steering required to approach the light source is equal in both left, as well as in both right positions. The distance the robot must travel is equal in both environments where the light source is close, as well as in both environments where the light source is far. The exact positions of the light source were determined as follows. The robot was simulated for the same amount of time steps as in the experiments, but its steering motor was manually set to its leftmost extant of \(-45^\circ\). The propulsion motor was set to its fastest setting. The light source was then placed in environment LF where the robot was at the last time step. Analogously, the placement was done in the remaining environments. Due to the different morphologies of the wheeled and the legged robots, the light source was positioned differently for the two robots.

3.2 The Robot Bodies

Experiments were performed on two different robot bodies. One is a tricycle, shown in Fig. 3.2a with two independently controlled degrees of freedom: the speed and the steering angle of the front wheel. The other body is a legged robot as shown in Fig. 3.2b. The legged robot features two limbs, one is attached to each of the two segments of the main trunk. To minimize the number of joints needed for locomotion and simplify the control needed, the limbs are comprised of two plates with different friction coefficients. The front plate of each leg (the black bars in Fig. 3.2b) has a lower friction coefficient than the rear plate (grey). This causes the robot to push itself forward when the limbs are moved backward and the high-friction plate touches the ground. When the limbs move

\(^1\)ODE is an open source physically realistic simulator, freely available for download at [http://www.ode.org/](http://www.ode.org/)
forward, the low friction plate slides on the ground and backward motion is impeded. Propulsion can therefore be achieved by an oscillation of the limbs’ tilt angles, which are moved in sync\(^2\). Steering is achieved by actuation of a hinge joint in the main trunk allowing turning within the coronal plane. The robot moves to the left if the front body rotates to the right and vice versa. This counter-intuitive behavior results from the particular morphology and is of no importance to the results of the experiment.

Both body plans allow a near complete separation of steering and propulsion. This is essential, as it leads to modularly varying subgoals: Moving to the far left light source requires the same amount of propulsion than moving to the far right light source. Similarly, moving to the far left light source, requires the same steering motion as moving to the close left light source. Each environment is therefore composed of two subgoals, i.e. finding the appropriate motion for both degrees of freedom. Two environments always have one of these subgoals in common.

The major difference between the two body plans is that a controller for the legged robot must apply oscillating torque to the leg motors to reach the light source. A controller for the wheeled robot must apply a constant speed and steering angle.

### 3.3 The Controllers

The neural networks used to control the robot are encoded as autonomous random boolean networks (RBN), as introduced by Kauffman [13][14]. The desired value of each

\(^2\)A video of the legged robot’s gait is provided at [http://www.youtube.com/watch?v=CDRx2_7vXTE](http://www.youtube.com/watch?v=CDRx2_7vXTE)
degree of freedom was encoded by the binary activities of a constant number $R$ of nodes. If not stated otherwise $R$ was set to 2 allowing for 4 ($2^2$) different desired values for each degree of freedom. In addition to these nodes, either 0, 1 or 2 hidden nodes were included in the controller.

The nodes function as sensor nodes in the first time step of each simulation and as actuator nodes in all subsequent steps. This kind of short-term memory, previously explored in [23], allows for the encoding of the information about the environment into the network while keeping the network dynamically autonomous (i.e. not influenced by sensory input over time). Each environment corresponds to a unique set of initial node activities, as shown in the first row of table 3.1. The first bit always encodes the distance of the light source and the second bit encodes whether the light source is on the left or right. The third and fourth bit are identical to the first and second bit, respectively, providing the information to both motors: in subsequent time steps, bits 1 and 2 determine the desired angle for motor 1; bits 3 and 4 determine the desired angle for motor 2. The node activities that lead to successful behavior, expressed in desired speed and desired steering angle for the wheeled robot, are shown in the third and fourth rows of table 3.1.

In the legged case, the initial encoding of the environment is the same, but the controller must settle into a cyclic attractor that leads to the motions required to reach the light source. The turning rate of the robot is directly dependent on the desired angle.

### Table 3.1: Encoding of the initial conditions and the final node activities for the wheeled robot that, if reached by a controller, would result in the robot reaching the light source at that position. The last row shows how the node activities are translated into desired speed and desired steering angle.

<table>
<thead>
<tr>
<th>Environment</th>
<th>LC</th>
<th>RC</th>
<th>LF</th>
<th>RF</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initial Activity</td>
<td>$-1 -1/ -1 -1$</td>
<td>$-1 +1/ -1 +1$</td>
<td>$+1 -1/ +1 -1$</td>
<td>$+1 +1/ +1 +1$</td>
</tr>
<tr>
<td>Final Activity</td>
<td>$-1 -1/ -1 -1$</td>
<td>$-1 -1/ +1 +1$</td>
<td>$+1 +1/ -1 -1$</td>
<td>$+1 +1/ +1 +1$</td>
</tr>
<tr>
<td>Desired Values</td>
<td>$15/ -45^\circ$</td>
<td>$15/45^\circ$</td>
<td>$45/ -45^\circ$</td>
<td>$45/45^\circ$</td>
</tr>
</tbody>
</table>
sent to the spine. Varying the speed can be achieved by either varying the frequency of the oscillation in the leg movement, the amplitude of the oscillation, or both. There are multiple solutions leading to success, as the same activity pattern can be achieved by different controllers. Additionally, multiple activity patterns can lead to success, as a goal is considered to be achieved if the robot is within a circle around the light source. How the node activities translate into desired leg angle and turning angle is shown in table 3.2.

3.4 The Evolutionary Algorithm

The weights of the neural networks were optimized using an evolutionary algorithm that employs elitism. Each evolutionary run was initialized with an initial population of 16 random controllers, after which each of the controllers was evaluated. The next generation of controllers contained the best controller of the current generation as well as mutated copies of the best four controllers of the current generation. The fitness function for the legged robot was:

\[
f_L = \prod_{E \in \mathcal{E}} \left(1 - \frac{d_E}{2D_E}\right)
\]

where \(d_E\) is the distance of the robot to the light source in environment \(E\) at the final time step and \(D_E\) the distance of the robot to the light source in environment \(E\) at time step 0. \(\mathcal{E}\) is the set of the environments the controller was evaluated in. A controller was considered successful if the fitness was greater than 0.8. The best case of 1 can be achieved if the robot reaches the exact position of the light source at the final time step in all evaluated environments. The worst case of 0 corresponds to a displacement of the initial distance between the robot and the light source in the opposite direction.

In the wheeled case the distance-dependent fitness was multiplied by a factor that penalizes the number of time steps \(t_s\) that it takes until the controller settles into a fixed point attractor:

\[
f_W = \begin{cases} 
\prod_{E \in \mathcal{E}} \left(1 - \frac{d_E}{2D_E}\right) \cdot \left(1 - \frac{t_s - 1}{N}\right), & \text{if } t_s \leq N \\
0, & \text{if } t_s > N
\end{cases}
\]

\(N\) is the number of nodes in the network (\(N = 4\) in all experiments on the wheeled
robot). It is reasonable to use the number of nodes as the upper limit for the settling time, since the expected settling time is proportional to the number of nodes.

Shaping is a method to incrementally approach the final goal by selecting controllers for a simpler task first and then gradually increase the complexity of the task [5, 10, 17]. A shaping schedule was implemented as follows. The controllers were first evaluated only in one environment. As soon as a successful controller was found for this environment, the next environment was introduced and all controllers in the evaluated population were evaluated in both environments. Once success was achieved in both environments, the controllers were re-evaluated in three environments and finally in four environments. The experiment was successful if a controller was found that succeeded in all four environments. Otherwise, the experiment was aborted after a maximum number of 5000 generations or 5 hours of computation time. If not stated otherwise, the order of environments was LC, RF, RC, LF, which alternates close and far positions of the light source. This order was chosen as it maximizes the number of switches from one environment to another that vary both the distance (from close to far or vice versa) and the direction (from left to right or vice versa).

Mutation was performed using the biased mutation operator originally formulated by Espinosa-Soto and Wagner [8]. The probability that a node is mutated is set to $p_m = 1/N^2$, where $N$ is the total number of nodes and $N^2$ thus the number of possible connections. Once a node is selected to undergo mutation, the probability of losing a connection is defined as

$$p(u) = \frac{4r_u}{4r_u + (N - r_u)} \quad (3.3)$$

and the probability of acquiring an additional connection as $q(u) = 1 - p(u)$. $r_u$ represents the number of incoming connections to node $u$ (the number of regulators, following the terminology of [8]). The connection that is removed or added is chosen randomly out of all incoming connections to node $u$ or out of all the pairs of nodes that are not yet connected, respectively. The biased mutation operator ensured that the number of incoming connections of each node was in the range of 2 to 4. According to [1] a degree of 2 allows for maximum evolvability and stability.

### 3.5 Modularity

It is common in the literature (e.g. [12, 8]) to define the modularity of a network under a certain partition according to Girvan and Newman [9, 18], as

$$Q = \sum_{i \in M} \left[ \frac{l_i}{L} - \left( \frac{d_i}{2L} \right)^2 \right] \quad (3.4)$$

where $M$ is the set of modules of the partition; $L$ is the total number of edges in the network; $l_i$ represents the number of edges within module $i$ and $d_i$ is the sum of
the number of connections that each node in module $i$ has. $Q$ is usually normalized as follows

$$Q_n = \frac{Q - Q_{ran}}{Q_{max} - Q_{ran}}$$

(3.5)

where $Q$ is calculated as described above in equation 3.4; $Q_{ran}$ is the average modularity of 100,000 randomly generated networks with the same number of nodes and the same connectivity distribution and $Q_{max}$ the maximum possible modularity of a network with these properties. $Q_n$ of an averagely modular network is then 0; $Q_n = 1$ means that the network features maximum modularity possible; a negative $Q_n$ corresponds to a modularity below average.

The final modularity of a controller is defined as the maximum $Q_n$ of all possible partitions. In this work only partitions that separate the nodes into driving nodes and steering nodes are considered, i.e. in the case of zero hidden nodes, only one partition was considered that allocates node 1 and 2 to the propulsion module and node 3 and 4 to the steering module. In the case of one hidden node, three different partitions were considered: the hidden node as a separate third module, the hidden node appropriated by the first module or the hidden node appropriated by the second module. Similarly, there are nine different partitions in the case of two hidden nodes.

The mean modularity was calculated at each evolutionary stage as the mean modularity of a sample of controllers that were successful and unique at that stage. The sample size in each experiment was the minimum number of unique successful controllers of all stages in that experiment. As many runs were conducted as was necessary to get a minimum sample size of at least 30, which is why the number of runs executed differs for each experiment.

To test the statistical significance of the increase of modularity at a certain evolutionary stage compared to random controllers, the independent two-sample Student’s t-test was used to compare a sample at that stage to a sample of initial random controllers.
4 Results

In this work, a total of 16 experiments were performed. Each experiment is comprised of multiple independent runs initialized with different random networks. An overview of the experiments is provided in table 4.1. Experiments 1 to 4 were conducted in physically realistic simulation on the wheeled or legged robot. On the wheeled robot, no hidden nodes were included in the controllers. On the legged robot one or two hidden nodes were used. Experiments 5 to 13 were conducted without simulation: cyclic attractors with different amplitudes or frequencies were directly selected for in one part of the network and fixed point attractors were directly selected for in the other part. Random walks were conducted on controllers with 0, 1 or 2 hidden nodes in experiments 14 to 16. Due to the different success rates of the experiments, the number of executed runs widely differs from 200 for easier tasks to 3000 for the seemingly more difficult task in experiment 4.

Sections 4.1 and 4.2 show the results of the experiments performed in physically realistic simulation on the wheeled robot and the legged robot, respectively. In section 4.3 the results of the runs without simulation are provided. The results using random walks are reported in the last subsection.

<table>
<thead>
<tr>
<th>Exp.</th>
<th>Robot</th>
<th>Nodes</th>
<th>Fitness</th>
<th>Environments</th>
<th>Runs</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Wheeled</td>
<td>4</td>
<td>Distance &amp; settling time</td>
<td>LC, RF, RC, LF</td>
<td>200</td>
</tr>
<tr>
<td>2</td>
<td>Legged</td>
<td>5</td>
<td>Distance</td>
<td>LC, RF, RC, LF</td>
<td>2500</td>
</tr>
<tr>
<td>3</td>
<td>Legged</td>
<td>6</td>
<td>Distance</td>
<td>LC, RF, RC, LF</td>
<td>1500</td>
</tr>
<tr>
<td>4</td>
<td>Legged</td>
<td>6</td>
<td>Distance</td>
<td>LC, RC, LF, RF</td>
<td>3000</td>
</tr>
<tr>
<td>5</td>
<td>None</td>
<td>6</td>
<td>Varying amplitude</td>
<td>None</td>
<td>200</td>
</tr>
<tr>
<td>6</td>
<td>None</td>
<td>10</td>
<td>Varying amplitude</td>
<td>None</td>
<td>200</td>
</tr>
<tr>
<td>7</td>
<td>None</td>
<td>14</td>
<td>Varying amplitude</td>
<td>None</td>
<td>500</td>
</tr>
<tr>
<td>8</td>
<td>None</td>
<td>6</td>
<td>Varying frequency</td>
<td>None</td>
<td>200</td>
</tr>
<tr>
<td>9</td>
<td>None</td>
<td>10</td>
<td>Varying frequency</td>
<td>None</td>
<td>200</td>
</tr>
<tr>
<td>10</td>
<td>None</td>
<td>14</td>
<td>Varying frequency</td>
<td>None</td>
<td>200</td>
</tr>
<tr>
<td>11</td>
<td>None</td>
<td>6</td>
<td>Varying frequency &amp; oscillation</td>
<td>None</td>
<td>200</td>
</tr>
<tr>
<td>12</td>
<td>None</td>
<td>10</td>
<td>Varying frequency &amp; oscillation</td>
<td>None</td>
<td>200</td>
</tr>
<tr>
<td>13</td>
<td>None</td>
<td>14</td>
<td>Varying frequency &amp; oscillation</td>
<td>None</td>
<td>500</td>
</tr>
<tr>
<td>14</td>
<td>None</td>
<td>6</td>
<td>None (random selection)</td>
<td>None</td>
<td>30</td>
</tr>
<tr>
<td>15</td>
<td>None</td>
<td>10</td>
<td>None (random selection)</td>
<td>None</td>
<td>30</td>
</tr>
<tr>
<td>16</td>
<td>None</td>
<td>14</td>
<td>None (random selection)</td>
<td>None</td>
<td>30</td>
</tr>
</tbody>
</table>

Table 4.1: Overview of the Experiments
4.1 Wheeled Robot

The first part of this section provides detailed information about one typical evolved controller for the wheeled robot that succeeds in all four environments. The remainder of the section reports the overall results of the experiments on the wheeled robot.

Since the architecture of a controller is fixed for each experiment, a controller is uniquely identified by the weight matrix encoding the connections of the controller. A weight can either be $-1$ for an inhibitory connection, $1$ for an excitatory connection, or $0$ if no connection between the corresponding two nodes is present. The weight matrix of a typical successful controller evolved for the wheeled robot is

$$ W = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (4.1) $$

In this work, the controllers are visualized using two representations of the weight matrices. The controller for the wheeled robot\footnote{A video of the simulated wheeled robot controlled by this network is provided at http://www.youtube.com/watch?v=bU38J7UQ4Cg.} represented by the weight matrix $W$ is shown in Fig. 4.1a. On the left it is represented by a directed graph, where the nodes are plotted as points and the connections as arrows or circles around the nodes if they are recurrent connections. On the right the weights are encoded as colored squares arranged in the same manner as the weight matrix. In this representation each square corresponds to one element of the weight matrix. As an example, the element in row 2 and column 1 is an excitatory connection from node 1 to node 2. Consequently, the blocks on the primary diagonal ([1:2, 1:2] and [3:4, 3:4]) contain intra-module connections, and the blocks on the secondary diagonal ([1:2, 3:4] and [3:4, 1:2]) contain inter-module connections. As can be seen, there is no connection between the two modules in this particular example. Hence, this controller is highly modular, but it does not reach the maximum modularity.

The fitness of the example controller during its evaluation in all four environments can be seen in Fig. 4.2a. This plot neglects the term of the fitness function that is dependent on when a controller settled into a fixed point attractor. It only shows the distance-dependent term of the fitness in the current environment, as defined in equation 3.1.

The desired values generated by the controller, which lead to the fitness reported in Fig. 4.2a are shown in Fig. 4.2b. A speed of 15 means slowly forward, 45 quickly forward and the negative values the same backward. The steering angle is stated in degrees; negative values correspond to a left turn and positive values to a right turn. The controller always settles into a fixed point attractor during the first time step. This is required, as the fitness threshold that defines success is 0.8 and a settling time of 2 time steps already decreases the maximum possible fitness to 0.5.

Fig. 4.1b shows the average connection weights of all unique successful controllers evolved on the wheeled robot. Only the existence of a connection was considered, not...
whether it was an inhibitory or excitatory connection. The thicker a line or darker a matrix element, respectively, the more often the corresponding connection occurred. In the directed graph, each line shows the average of the corresponding two possible directed connections between the respective two nodes. It can be seen that connections within the two modules are more frequent than connections between the two modules. This demonstrates modularity in an intuitive way.

In experiment 1, 200 evolutionary runs were executed using the standard shaping schedule LC, RF, LF, RC and no hidden nodes. Each environment was simulated for 200 time steps. The light source was not arbitrarily positioned: Since a controller must settle into a fixed point attractor, the light source has to be positioned such that the robot can reach it without changing the speed or steering angle after its controller once settled.

The results of the experiments on the wheeled robot are presented in Fig. 4.3 (black lines). The successes at each stage of the shaping schedule are shown in Fig. 4.3a. 114 (57% success rate) of these runs succeeded in all four environments. Due to the small controller size ($N = 4$) and the correspondingly small search space, some controllers were found multiple times. 55 of the 114 successful controllers were unique solutions. Mean modularity of the unique successful controllers started increasing after success in the first environment and continued increasing to a final value of 0.29 as shown in Fig. 4.3b. The difference in mean modularity of the successful controllers after three and then four environments were conquered compared to random controllers is statistically significant (p-value < 0.001).
Figure 4.2: Simulation results of a typical successful controller that settles into a fixed point attractor for the wheeled robot. Each environment was simulated for 200 time steps, after which the robot was moved back to the origin and the light source to the next location. (a) The controller’s fitness, as the robot moves in each of the four environments. The fitness is indirectly proportional to the distance of the robot to the light source. (b) The desired angles generated by the network during evaluation.

4.2 Legged Robot

Three experiments were executed on the legged robot: One experiment comprised of 2500 evolutionary runs using one hidden node; one experiment of 1500 evolutionary runs using two hidden nodes; and one experiment of 3000 evolutionary runs using an alternative left/right alternating shaping schedule LC, RC, LF, RF and two hidden nodes. Each environment was simulated for 1000 time steps. Here, the light source was positioned using a similar method as in the wheeled robot case: Actuations that made the robot move slowly to the left were applied and the position where the robot was after 1000 time steps was used to place the light source in environment LC. Actuations that made the robot move quickly to the left were applied to determine the position of the light source in environment LF. The same was done for the two remaining environments RC and RF.

The results are shown along with the results of the wheeled case in Fig. 4.3. The blue lines represent the experiment with one hidden node; the green lines represent the
Successful Runs

Modularity

Figure 4.3: Results from evolving controllers to succeed first in one environment, then in two environments, then in three environments and finally in four environments. Black: Wheeled robot, no hidden nodes; blue: legged robot, 1 hidden node; green: legged robot, 2 hidden nodes; red: legged robot, 2 hidden nodes, left/right alternating shaping order. (a) Number of runs that produced successful controllers for each evolutionary stage. (b) Modularity of the unique successful controllers at the final stage compared to the modularity of the initial random controllers is significantly greater (p-value < 0.001 in all four series). Thick lines: mean modularity; dotted lines: one unit of standard error of the mean.

A typical successful controller for the legged robot with two hidden nodes and evolved in a run with standard shaping order (LC, RF, LF, RC) is described in more detail in the remainder of this section.

The example controller is shown in Fig. 4.4 and exhibits a high modularity that is achieved by assigning both hidden nodes to the module responsible for propulsion (blue shaded). Node 6, the second hidden node, is particularly interesting as it displays no incoming connections, hence, is used as a bias node.

Fig. 4.5 shows the fitness and the desired angles generated by the controller while the robot is moving in each environment. It is apparent in Fig. 4.5b that in this

---

A video of the simulated legged robot controlled by this network is provided at http://www.youtube.com/watch?v=sCuUtGQsTsw
Figure 4.4: The network underlying the behavior reported in Fig. 4.5. Blue: nodes responsible for propulsion; yellow: nodes responsible for steering; green: excitatory connections; red: inhibitory connections; self-connections are drawn as circles around the nodes. The hidden nodes are white. In this particular controller, both hidden nodes are used by the module responsible for propulsion. On the right, an alternative visualization of the same controller. The element in the \( i \)-th row and \( j \)-th column shows the connection from node \( j \) to node \( i \). Nodes 1 and 2 encode propulsion, nodes 3 and 4 are the hidden nodes, and nodes 5 and 6 encode steering.

particular controller the two different distances required to approach the close and the far light sources are achieved by varying the frequency of the oscillatory leg movement. The low frequency corresponds to the close placements (time steps 0 to 1000 and 2000 to 3000) and the high frequency, allowing a faster forward speed, corresponds to the far placements (remaining intervals). Due to the teetering motion of the body, caused by the walking gait, which influences the distance to the light source, the leg oscillation frequency is also visible in the fitness in Fig. 4.5a. The robot needs in this case eight gait cycles to reach the light source in the close placements and twice as many to reach the light source in the far placements. The varying amplitude of the fitness oscillation is not related to an actual change in amplitude in the leg oscillation, but rather is caused by the normalization of the distance to the light source; the teetering motion is simply greater compared to the distance if the light source is closer. This particular controller settles into a cyclic attractor within one time step, but in contrast to the wheeled case, here, this is either accidental or beneficial for the controller, but is not explicitly rewarded.

The average connectivity density in all unique successful controllers, for all three experiments on the legged robot, are shown in Fig. 4.6. An insight into the final modularity can be gained from the left images: The inter-module connections reported there are in general less pronounced than the intra-module connections. However, the degree of modularity is not clearly visible by eye, as the recurrent connections influence modularity and do not well support the seeming rise of modularity in the directed graphs. A better intuition is provided by the second visualization: There is a higher average connectivity in the blocks on the primary diagonal ([1:4, 1:4], [5:6, 5:6]) than in the residual blocks, which corresponds to more intra-module connections compared to inter-module connections.
Fitness of a Successful Controller for the Legged Robot

Desired Angles Generated by a Successful Controller for the Legged Robot

Figure 4.5: Simulation results of a typical successful run that settles into a cyclic attractor for the legged robot. (a) The controller’s fitness, as the robot moves in each of the four environments. The fitness is indirectly proportional to the distance of the robot to the light source. (b) The desired angles applied in the simulation. The desired angles are functions of the binary node activities. Each environment was simulated for 1000 time steps, after which the robot is moved back to the origin and the light source to the next location.

4.3 Explicit Selection of Cyclic Activity Patterns

Nine experiments were conducted without the simulator. Instead, particular dynamic patterns observed in the networks evolved on the legged robot were selected for. More specifically, cyclic attractors were selected for in one part of the network (nodes 1 to \( R \), where \( N = 2R + 2 \) and nodes 1 to \( R \) encode the desired angle of motor 1), and a particular fixed point attractor was selected for in the other part of the network (nodes \( R + 3 \) to \( 2R + 2 \) that encode motor 2). The two hidden nodes (nodes \( R + 1 \) and \( R + 2 \)) were free to be appropriated by either (or neither) of the modules. As in the simulated experiments on the robots, the controllers were first initialized with one initial condition (set of initial node activities) and evaluated. Upon success, they were consecutively evaluated with two initial conditions, then with three initial conditions and finally with four initial conditions. After initialization, the node activities were updated for a certain number of time steps linearly dependent on \( N \). A fitness value was assigned for each of
Figure 4.6: The average connectivity of all unique successful controllers evolved for the legged robot. The thicker a line or darker a square, respectively, the more connections (inhibitory or excitatory) occurred on average. In the right images, the intra-module connections are in the blocks on the primary diagonal ([1:3, 1:3], [4:5, 4:5] in (a); [1:4, 1:4], [5:6, 5:6] in (b,c)) and the inter-module connections in the remaining blocks. It can be seen that the intra-module connections are more pronounced than the inter-module connections in all three cases.

the four evaluations with different initial conditions, and the final fitness was, as in the other experiments, the product of the four partial fitness values achieved with each of the four initial conditions. Instead of evaluating the networks based on their influence on the robot behavior in the simulation, here, desired network dynamics were selected for directly as described in the following paragraphs.

In the first three experiments (5 to 7) the set of nodes selected for settling into a cyclic
attractor was additionally required to acquire two different amplitudes depending on the initial conditions. This corresponds to one of the two observed behaviors of the legged robot, where two different distances were reached by varying the oscillation amplitude of the legs. Controllers in experiment 5 contained 6 nodes: 2 nodes for each motor and 2 hidden nodes; controllers in experiment 6 contained 10 nodes: 4 nodes for each motor and 2 hidden nodes; controllers in experiment 7 contained 14 nodes: 6 nodes for each motor and 2 hidden nodes.

To illustrate what kind of activity pattern is selected for, the desired values generated by a successful controller ($N = 10$) that exhibits two different amplitudes are shown in Fig. 4.7a. It can be seen that in the first 80 time steps the amplitude of the leg angle is smaller than it is in time steps 80 to 160. It was observed in the experiments on the legged robot that a smaller amplitude in the leg angle oscillation leads to a slower forward motion. This controller therefore could succeed when applied to the legged robot and the environments were presented in the order LC, RC, LF, RF. However, because the offset of the oscillation and the frequency are not specified the dynamics may not necessarily correspond to a successful approach of the light source.

The fitness function that was used to generate that activity pattern rewards the successful settling into the right fixed point attractor for nodes $R + 3$ to $2R + 2$, which encode motor 2, by calculating the Hamming distance between the desired final node activities and the actual final node activities. This value is then halved for each of the following conditions that was not met: First, the period of the activity pattern of nodes 1 to $R$, which encode motor 1, is equal to 2, meaning that there is oscillation in motor 1 (the frequency was fixed to 2 to simplify the comparison of the amplitude). Second, nodes $R + 3$ to $2R + 2$, which encode motor 2, must feature a period of 1, i.e. remain in a fixed point attractor. Finally, the amplitude of the oscillation must be equal in initial conditions 1 and 2, greater in initial condition 3 and equal to the greater of the two amplitudes in initial condition 4.

The average connections of all unique successful controllers that exhibit different amplitudes under these conditions are shown in Fig. 4.8. In all three cases it can be seen that there are more connections within the nodes encoding the desired angle of motor 1 and within the nodes encoding the desired angle of motor 2 than there are between the two groups of motor neurons, thus indicating the evolution of modularity. For $N = 6$ and $N = 10$ the hidden nodes are more frequently used to support the oscillating motor 1; for $N = 14$ the affiliation of the hidden nodes varies in the runs. This is apparent in Fig. 4.8 as the hidden nodes are clearly denser connected to the propulsion nodes than to the steering nodes in the cases where $N = 6$ and $N = 10$. In the case where $N = 14$, however, this is not the case. A bar chart showing the average affiliation of the hidden nodes in this experiment is provided in the discussion section in Fig. 5.3.

The number of successful controllers and the mean modularity of the controllers selected for varying amplitude are shown in Fig. 4.9. The success rate was much lower with $N = 14$ and therefore more runs were needed to obtain at least 30 successful runs. Modularity increased in all three experiments statistically significantly (p-value < 0.001). This suggests that modularity will evolve independent of the number of nodes per motor when selection is applied as described above.
In experiments 8 to 13 the dynamics of the nodes that encode the desired angle of motor 1, and that were selected for settling into a cyclic attractor, were here selected to acquire two different frequencies (as opposed to amplitudes as in experiments 5 to 7) depending on the initial conditions. This is the second variant observed on the legged robot that generated different walking speeds. Three experiments (8 to 10) with $N = 6$, $N = 10$ and $N = 14$ were conducted. Three further experiments (11 to 13, again with $N = 6$, $N = 10$ and $N = 14$) were conducted where not only oscillation in the desired angle encoded in the activities of nodes 1 to $R$ was selected for, but also each single activity of these nodes was required to oscillate.

The desired values generated by a successful controller ($N = 10$) that exhibits two different frequencies are shown in Fig. 4.7. It can be seen that the frequency of the oscillation in motor 1 is lower in time steps 1 to 80 as it is in the remaining time steps.
Figure 4.8: The average connectivity of all unique successful controllers exhibiting different amplitudes. The thicker a line or darker a square, respectively, the more connections (inhibitory or excitatory) occurred on average. In the right images, the intra-module connections are in the blocks on the primary diagonal ([1:4, 1:4], [5:6, 5:6] in (a); [1:6, 1:6], [7:10, 7:10] in (b); [1:8, 1:8], [9:14, 9:14] in (c)) and the inter-module connections in the remaining blocks. It can again be seen that the intra-module connections are more pronounced than the inter-module connections in all three cases.

Again, the networks evolved in this experiment could therefore be successful if applied to the legged robot, but not necessarily, as the offset and amplitude of the oscillations are not specified and the robot therefore not necessarily walks to the light source.

Here, the fitness function rewarded for settling into the right fixed point attractor of the nodes that encoded motor 2 as in experiments 5 to 7. The additional conditions that caused the fitness to be halved, if not met, are as follows: First, in initial condition 1, the period of the node activities encoding motor 1 must be greater than 2 (cyclic attractor)
Successful Runs when Selecting for Varying Amplitude

Modularity when Selecting for Varying Amplitude

Figure 4.9: Results from evolving controllers to succeed first in one initial condition, then in two initial conditions, then in three initial conditions and finally in four initial conditions. One module of the network was explicitly selected to settle into a cyclic attractor with varying amplitude depending on the initial conditions and the other module to settle into a fixed point attractor. Blue: $N = 6$; green: $N = 10$; red: $N = 14$. Modularity increased statistically significantly (p-value < 0.001) in all experiments and independently of the number of nodes.

and in initial condition 2, the period must be equal. In initial condition 3, the period of oscillation must be longer than in initial conditions 1 and 2, and finally, in initial condition 4 it must be equal to the period in initial condition 3. In experiments 11 to 13, each single node activity encoding the desired angle of motor 1 must also oscillate (period > 1).

The average connections of all unique successful controllers that exhibit different frequencies are shown in Fig. 4.10. For $N = 6$ modularity is pronounced in both experiments (8 and 11) to about the same amount. For $N = 10$ it can be noted that modularity is greater in the case where all nodes encoding motor 1 are required to oscillate (Experiment 12). For $N = 14$ modularity is weakly pronounced in experiment 10, but obvious when all nodes encoding motor 1 are required to oscillate (Fig. 4.10f). As in the experiments with varying amplitude, the affiliation of the hidden nodes is less pronounced, as more nodes are included in the controller.

The number of successful controllers and the mean modularity of the controllers selected for varying frequency are shown in Fig. 4.11. Modularity increased in all six experiments statistically significantly (p-value < 0.001). However, in experiments 8 to 10, where not all nodes of the oscillating module were required to oscillate, it can be seen that the increase of modularity is lower, as more nodes are used (Fig. 4.11b). This trend is not apparent when all nodes encoding motor 1 are required to oscillate (Fig. 4.11d).
4.4 Benchmark

For each of three different controller topologies used in the other experiments ($N = 6$, $N = 10$, $N = 14$), one experiment of 30 evolutionary runs was conducted with random selection: Instead of applying elitism to create offspring, a random subset of each generation was chosen to form the parents of the next generation. Mean modularity was always calculated after 500 generations evolved, until 5000 generations in total were evolved. As can be seen in Fig. 4.12 modularity stays within the interval $[-0.1, 0.1]$ and
Successful Runs when Selecting for Varying Frequency

Modularity when Selecting for Varying Frequency

Successful Runs when Selecting for Varying Frequency and Oscillation in All Nodes of the Cyclic Module

Modularity when Selecting for Varying Frequency and Oscillation in All Nodes of the Cyclic Module

Figure 4.11: (a, b) Results from evolving controllers to succeed first in one initial condition, then in two initial conditions, then in three initial conditions and finally in four initial conditions. One set of nodes was explicitly selected for settling into a cyclic attractor with varying frequency depending on the initial conditions and the remaining nodes for settling into a fixed point attractor. Blue: $N = 6$; green: $N = 10$; red: $N = 14$. Modularity increase is statistically significant (p-value < 0.001) in all experiments, but less pronounced with more nodes. (c, d) Here, each node in the oscillating module was required to oscillate. Modularity increase is independent of the number of nodes encoding each motor.
Figure 4.12: Modularity does not increase if evolution is performed without a goal, i.e. parents are chosen randomly rather than according to their fitness. Blue: \(N = 6\); green: \(N = 10\); red: \(N = 14\).

thus no significant increase in modularity towards later generations can be observed.

These experiments served as a benchmark to compare the development of modularity of the evolved controllers to the development of modularity in random walks. They ensure that modularity did not emerge as the result of some intrinsic bias of the evolutionary algorithm used, but rather because modular neural networks were beneficial for the robot to succeed at the task at hand.
5 Discussion and Analysis

Experiments were performed in physically realistic simulation on a wheeled robot, on a legged robot and without simulation. Controllers for the wheeled and legged robots were selected by an evolutionary algorithm for their ability to cause taxis behavior in the robots. The wheeled robot could accomplish this using a controller that fell into a fixed point attractor such that constant torque was applied to the wheels. The legged robot could accomplish this using a controller that fell into a periodic attractor in which oscillations in torque were applied to the legs. For this reason, experiments on the wheeled robot were used to investigate the evolution of modularity when selecting for fixed point attractors and on the legged robot to investigate the evolution of modularity when selecting for cyclic attractors.

Even the relatively simple tasks described within this paper show a great discrepancy between success rates in different experiments. Attaining one particular fixed point attractor for each of the four initial conditions in the wheeled case seems to be a significantly easier task than attaining one of the cyclic attractors that leads the legged robot to the light source. The number of hidden nodes and the shaping schedule used play a considerable role in the evolvability of a network to succeed in a given task.

By reference to particular experiments, some factors that influence evolvability are discussed in the following part of this section. In the remainder of the section, further insights into what causes the evolution of modularity are shared and limitations of this work identified. Finally, questions remaining open are stated to indicate possible directions for future work.

5.1 Selection for Fixed Point Attractors

What is essentially asked for in the experiment on the wheeled robot is a neural network \( N = 4 \) that, depending on four different initial conditions, settles into four specific fixed point attractors always within one time step. In other words, 4 out of 16 different possible initial conditions each have to be matched to one of 16 possible final conditions. The specific initial conditions and desired final node activities used in the experiment on the wheeled robot are summarized in table 3.1. The same experiment could also be done with any combination of four initial and four final conditions. In this work it was also done using a second encoding provided in table 5.1 and otherwise identical conditions as in the experiment on the wheeled robot that was described above.

Of the 200 runs using this encoding no controller evolved that succeeded in more than three environments. The values used in this experiment on the wheeled robot required much more bits to flip from the initial conditions to the desired final values than the experiment described beforehand. Therefore, it is possible, that no network exists that
matches these four initial conditions to fixed attractors with the desired outputs and additional nodes would be required to achieve that. Even when the limitation on the settling time was relaxed no success was recorded. This shows that the encoding highly influences the feasibility of a task.

### 5.2 Selection for Cyclic Attractors

In all experiments using the simulator, the light source was placed at two different distances with respect to the robot, and the number of simulated time steps was constant in all environments. A successful controller therefore had to be able to perform dynamics that result in two different speeds for the robot to reach both the close and the far light sources. On the wheeled robot, this simply corresponds to different values of the fixed point attractor, as the torque and consequently the speed is directly controlled. On the legged robot two different speeds can be accomplished by two different frequencies in the leg angle oscillation or by two different amplitudes (or by combinations of both). It is shown in Fig. [5.1] how the controllers achieved this speed variance in all three experiments on the legged robot. In all experiments, most controllers exhibited a variation in frequency or a variation in both frequency and amplitude. If only one hidden node was included in the controller, no evolved solution exhibited a variation in amplitude only.

Controllers were also evolved without a simulator, but in an additional experiment where selection for attaining different amplitudes was applied (same conditions as in experiment 5, but with only one hidden node) were able to show amplitude variation while attaining the desired fixed point attractor in motor 2. Successful controllers of this experiment cause locomotion with two different speeds when applied to the legged robot. However, since offset and frequency of the oscillation were not specified, the motion caused by these controllers do not drive the robot to the exact position of the light source. Nevertheless, this raises the assumption that controllers with 5 nodes that exhibit the desired behavior exist, but are difficult to evolve on the robot. In the case of controllers with 6 hidden nodes, controllers that exhibit amplitude variation definitely exist, but are more difficult to evolve than frequency variation.

A reason why frequency variation is preferred may have been the positioning of the light source. The light source in the close placements was placed by applying oscillations in the desired leg angles that had a lower frequency than the oscillations applied to place the light source in the far placements. A controller causing a leg angle oscillation that

<table>
<thead>
<tr>
<th>Environment</th>
<th>LC</th>
<th>RC</th>
<th>LF</th>
<th>RF</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initial Activity</td>
<td>+1 +1/ −1 +1</td>
<td>−1 +1/ −1 +1</td>
<td>+1 −1/ −1 −1</td>
<td>−1 −1/ −1 −1</td>
</tr>
<tr>
<td>Final Activity</td>
<td>−1 +1/ −1 −1</td>
<td>−1 +1/ +1 +1</td>
<td>−1 −1/ −1 −1</td>
<td>−1 −1/ +1 +1</td>
</tr>
<tr>
<td>Desired Values</td>
<td>15/ −45°</td>
<td>15/45°</td>
<td>45/ −45°</td>
<td>45/45°</td>
</tr>
</tbody>
</table>

Table 5.1: Alternative encoding of the initial conditions and the final node activities for the wheeled robot that, if reached by a controller, would result in the robot reaching the light source at that position. The last row shows how the node activities are translated into desired speed and desired steering angle.
drives the robot to a far target can therefore cause the robot to approach the close target by lowering the frequency in the leg angle oscillation. If however the oscillation would feature a smaller amplitude instead of a smaller frequency, the resulting robot speed would also result in a lower speed, but due to the robot morphology the travel direction of the robot is perturbed too. This suggests that controllers that cause variation in the speed by varying the oscillation amplitude are disadvantaged, as they perturb the travel direction of the robot. However, when the light source in the close placements was placed by applying oscillations in the desired leg angles that had a smaller amplitude (instead of frequency) than the oscillations applied to place the light source in the far placements, frequency variation was still the preferred method to achieve speed variation. The portion of controllers that exhibited amplitude variation did not change significantly. Consequently, the positioning of the light source can be neglected as a cause of this bias against amplitude variation, and varying amplitudes therefore must be harder to evolve on the legged robot than varying frequency.

The reason, instead, is presumably the low resolution of the desired angles (−45°, −15°, 15°, 45°). There are only a few intermediate steps between the two extremes, meaning that the amplitude can only be varied in big steps, and a controller either gets a very high fitness or a very low fitness, but cannot gradually improve. The frequency, however, can range from 2 to any value, hence, a slight change in frequency does not overly influence fitness. A controller can therefore gradually adapt from a frequency leading to success in the close environments to a frequency leading to success in the far environments. This hypothesis could be validated by increasing the resolution of the desired angles: Then, more intermediate steps between the two extreme desired angles could be encoded, and slighter changes in the oscillation amplitude were possible. In this

**Figure 5.1:** Relative occurrence of controllers that achieved speed variation by varying the amplitude, the frequency or both. A: varying amplitude, F: varying frequency, A&F: varying both amplitude and frequency. In all three experiments amplitude variation appeared less often. Using one hidden node, no controller at all varied the amplitude of the oscillation in the desired leg angle.
case, the portion of controllers that exhibit different amplitudes to accomplish different speeds should be greater, if the hypothesis is true.

Interestingly, in the experiments where network dynamics were directly selected for, the success rate in both cases (varying amplitude and varying frequency) seemed roughly equal. This is consistent with the stated hypothesis, as here a changing amplitude does not imply a highly perturbed fitness. Rather varying the amplitude does not influence the value of this fitness function to a greater extent than varying the frequency.

5.3 Hidden Nodes Affiliation

The experiments on the wheeled robot showed, that the steering could be achieved with just two nodes. Hence, it is not surprising that the hidden nodes in the legged case are more often appropriated by the module responsible for propulsion, since this module is required to show more complex dynamics. This is confirmed in Fig. 5.2 which shows the average node affiliations in all three experiments on the legged robot. The same is also apparent in Fig. 4.6. The average connections between the hidden nodes and the propulsion module (blue) are in general stronger than the connections between the hidden nodes and the steering module (yellow).

Fig. 4.8 and Fig. 4.10 show that this bias loses importance when more nodes are included in the controller. When only a few nodes are present, the hidden nodes are required to generate the desired dynamics (i.e. two different amplitudes depending on the initial conditions), however, when more nodes are present in the group that is required to exhibit oscillation, the hidden nodes lose importance and they are more equally distributed to both node groups.

5.4 Why Modularity Arises

Wagner [30] suggested that modularity could spontaneously arise as a result of stabilizing selection applied to one part of the network and directional selection applied to the other part. Lipson et al. [28, 16] suggested that modularity could spontaneously arise in tasks with modularly fluctuating subgoals. The task in this work is comprised of modularly fluctuating subgoals, as two environments always have either the required travel distance or direction in common. As a result, stabilizing selection was experienced in one part of the network while directional selection was experienced in the other part, e.g. when a controller was successful in environment LC and then was evaluated in both LC and LF, the steering part’s task stayed the same, but the propulsion part needed to adapt. Hence, in this case, stabilizing selection is applied to the steering part of the network and directional selection to the propulsion part.

The common hypothesis why modularity arises under these circumstances is that inter-module connections perturb the module that is under stabilizing selection [30]. Therefore, there is selection pressure against inter-module connections. To validate whether indeed this principle was causing the evolution of modularity in the experiments on the wheeled robot, an experiment with the same task but non-modularly varying subgoals
could be conducted. This could be achieved by increasing the range of approachable positions of the light source by using negative driving speeds and placing the light source at four positions that neither share direction nor distance with respect to the robot.

In the experiments in this thesis all information about the environments (left/right, close/far) is completely encoded in both groups of nodes that each encode a desired value. This implies that, while inter-module connections might be required to achieve the desired dynamics, they are certainly not required to spread the information in the network. Consequently, there is selection pressure against inter-module connections, and there might not be any advantage of these connections in the conducted experiments.

Sun and Deem [25] suggested that the increase of modularity is correlated to the rapidity and severity of environmental change. In the phototaxis task in this thesis, the environmental change can either be a change in direction (left/right) or a change in speed (close/far) or a change in both. On the legged robot, two different shaping orders were used, where the severity of the change between two consecutive environments is different: In the experiment with the left/right switching order (LC, RF, LF, RC) in two switches, both aspects (left/right and close/far) are changed and only in one switch, only one aspect is changed. In the experiment with the alternative shaping order (LC, RC, LF, RF), in only one switch, both aspects are changed. When comparing the evolution of modularity in these two cases after three and then four environments were conquered, a significant difference can be observed (p-value < 0.001): Fig. 4.3b shows that in experiment 4 (red line), where the environmental change is less severe, the evolution of modularity is less considerable than it is in experiment 3 (green line), where the environmental change is more severe.
Figure 5.3: Module affiliation of the hidden nodes when selecting for a cyclic attractor with varying amplitude in one part of the network and a fixed point attractor in the other part. When $N = 6$ at least one hidden node was typically appropriated by the oscillating module 1 (blue), the more nodes were included in the controller, the less important was the affiliation of the hidden nodes. With $N = 14$, the hidden nodes were in average equally distributed to both modules.

Besides modularity, evolvability is also greater if the environmental change is more severe: The experiment with less severe environmental change showed a success rate of roughly half the success rate of the same experiment with more severe environmental change. The shaping order should therefore be chosen carefully, as it influences evolvability and the evolution of modularity.

5.5 Scaling the Networks

All experiments where selection for certain network dynamics was applied were executed first with 6 nodes and then with 10 and 14 nodes to get an insight into how the results scale to larger networks. Instead of 2 nodes encoding each desired value, 4 or 6 nodes were used. The number of hidden nodes was fixed to 2 in all these experiments.

When experiment 8 (varying frequency) was repeated with more nodes in experiments 9 and 10, the increase of modularity (Fig. 4.11b) was less pronounced the more nodes were used to encode the desired angles. To investigate the reason for this, the activities of the nodes encoding the oscillating desired angle of motor 1 were observed in detail. To achieve an oscillation in the desired angle of motor 1, at least one of the two nodes was oscillating throughout all time steps, while the other node was either used to modulate the frequency in half of the time steps to form two different frequencies, it also oscillated throughout all time steps, or it stayed fix. In the latter two cases, the activity of this second node did not influence the fitness, as only the frequency and not the amplitude and the offset of the oscillation were considered.

When more nodes where used to encode the oscillating desired value, still only one
oscillating node was enough to cause the oscillation in the encoded value. The more
nodes to encode the desired angle that was required to oscillate were used, the greater was
therefore the portion of nodes that were not required to actively influence the behavior
and were allowed to show arbitrary dynamics.

Modularity supposedly evolved in experiment 8, as any connections from the group
of nodes that encoded the oscillating desired value to the group that encoded the fixed
desired value potentially interfered with the settling into a fixed attractor of the latter
group. When the number of nodes that encode the oscillation is greater than 2, however,
this group of nodes may consist mainly of nodes that attain a fixed attractor and only
few nodes that oscillate. The inter-module connections then do not heavily impede the
desired dynamics, hence, minimizing these connections is not necessarily beneficial, and
modularity is thus not required.

Concluding, it can be said, that the evolution of modularity due to selection for a
fixed point attractor in one part of the network and selection for a cyclic attractor in the
other part scales to larger networks if nodes are required to achieve the desired dynamics
independently of the number of nodes. Presumably, this is not only true in this case, but
also when other principles causing the spontaneous evolution of modularity are applied,
as idle nodes do not influence fitness thus no selection pressure against connections to
these nodes is applied.

5.6 Future Work

The networks considered in this work are comprised of four to six nodes when applied to
robots. An exception to this are the experiments where network dynamics were selected
for directly, and the networks had up to 14 nodes. These experiments investigated how
the results from the simpler networks used on the robots would scale up to larger net-
works and more complex robots. They show that it is important for the evolution of
modularity that there are no idle nodes, since those do not support the selection against
inter-module connections. To achieve the evolution of modularity in larger networks it is
therefore necessary to do experiments on more complex tasks that require more nodes,
but still feature some kind of separability of different traits. Since the success rate in
the rather simple tasks in this work is already relatively low, more advanced evolution-
ary algorithms, like hyperNEAT-LEO [29], an extension to the popular hyperNEAT
algorithm, could be used to evolve these more complex controllers.

With more complex tasks and controllers, eventually, there would be an advantage of
evolving networks with more than two modules. It may be possible to achieve that by
extending the concept of selecting for a fixed point attractor in one part of the controller
and for a periodic attractor in the other part to select for periodic attractors of different
frequencies in different parts of the controller.

Both robots used in this thesis are simple and can be built physically. Future work
could investigate whether modular controllers evolved on simulated robots work on phys-
ical robots. It would also be interesting to explore the evolution of modularity directly
on physical robots. It is to be expected, that the two degrees of freedom would not be
optimally separated as they are in simulation, and that noise would be included in the encoding of the environment. The degree of modularity in controllers evolved on physical robots might therefore be less severe compared to networks evolved on simulated robots. However, modularity evolved in these experiments due to multiple reasons, and it is to be expected that selection pressure towards modular neural networks on a physical system involving the same task would still be strong enough to cause the evolution of modularity.

Most important, once it comes to real world applications, is the switch to continuous perception of the environment. The method applied here, where the robot perceives the environment only in the first time step and then blindly moves based on that information, imposes a great limitation. Continuous sensor inputs, however, imply a change of architecture away from dynamically autonomous networks.
6 Conclusion

This work shows the spontaneous evolution of modularity in neural networks that control robots in a task involving locomotion towards a target. Modularity presumably evolved as a result of modularly varying subgoals that caused stabilizing selection to be applied to one part of the network and directional selection to be applied to the other part of the network. Experiments on a wheeled robot showed this on controllers that exhibited fixed point attractors; experiments on a legged robot showed this on controllers that exhibited cyclic attractors.

In a task that involved the selection for cyclic attractors in one part of the network and selection for fixed point attractors in the other part, it was investigated how the evolution of modularity scales from small networks of 6 nodes to larger networks of up to 14 nodes. It has been found that modularity evolves if most nodes are required for the achievement of the desired dynamics. If on the contrary many nodes are superfluous and are free to show arbitrary dynamics, there is less selective advantage of modular neural networks over non-modular neural networks. The absence of idle nodes in the networks might be an important condition for the evolution of modularity, as connections to these nodes do not influence fitness, hence, there is no selection pressure against these connections. Consequently, the complexity of the task should grow with the number of nodes in the controller. Future work should relax these limitations by gradually increasing the complexity of the controllers and the tasks. The challenge however is to maintain a certain degree of separability of the degrees of freedom, as otherwise it is difficult to design tasks that are comprised of modularly varying subgoals.

The limitations imposed on the task in this work are rather stringent and the controllers comprised of few nodes. However, multiple properties of the task were causing the evolution of modularity: thus, less restrictive limitations, more realistic conditions or even the physical implementation of the system should therefore still allow for the evolution of modularity. One of the most important limitations of the work done on the evolution of modularity in dynamically autonomous random boolean networks is the lack of the possibility to include sensory input. For real world applications continuous sensory input is needed; it is therefore required to extend this research to networks that process continuous sensory input.
Bibliography


