UNIVERSITEIT VAN AMSTERDAM

VU VRIJE UNIVERSITEIT AMSTERDAM

MSc Computer Science

Master Thesis

Evolving steerable pneumatic soft robots

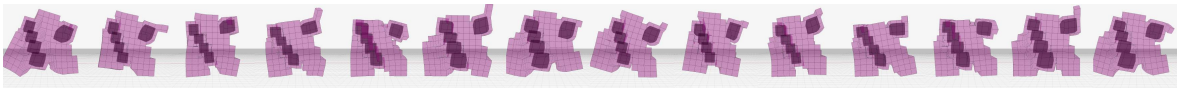
by

Thijs Meijerink

2661163 (VU), 10783857 (UvA)

Supervisors:	Prof. Dr. A.E. Eiben
	Dr. Karine Miras
Date:	Thursday 20th June, 2024

A thesis submitted in fulfillment of the requirements for the joint UvA-VU Master of Science degree in Computer Science



Abstract

Soft robotics is a young field that aims to develop a new type of robot made out of elastic, compliant materials. Due to these material properties, soft robots can work together with humans, and interact with soft objects or terrain, more effectively and safely than hard robots. Also, by offloading computational and control tasks to the dynamics of the body, these robots could be simpler and therefore cheaper.

However, the complex non-linear dynamics of soft materials make soft robots and their controllers challenging to design by hand. In answer to this, evolutionary robotics methods have been applied to virtual soft robots. Though promising, this work has at present several limitations. Firstly, evolved virtual soft robots are less realistic than their physical counterparts, limiting the practical applicability of the evolutionary methods. This goes especially for the actuation method, since existing work does not accurately model the pneumatic actuators that are used in most physical soft robots. Secondly, the lack of sensory apparatus in existing evolved soft robots severely limits the range of tasks soft robots might perform. Finally, controllers for soft robots used in the literature are often restricted to simple sine waves. More complex controllers are needed for different tasks, as well as to integrate sensing and actuation into a closed-loop controller.

To address these limitations, an evolutionary soft robotics system is proposed that extends existing work building on VoxCAD, a voxel-based soft robot simulator, and CPPN-NEAT, a method that leverages neuro-evolution to encode and evolve voxel-based soft robots. I extend the simulator with a realistic pneumatic actuation and sensing model. The CPPN-based robot body encoding is extended with a post-processing step to enable the effective encoding of pneumatic chambers. Finally, a simple recurrent neural network controller is implemented.

Using this system, robots are evolved for two tasks, gait learning and directed locomotion, to answer research questions pertaining to the impact of pneumatic sensing and actuation methods on the evolved robots, and the division of tasks between controller and morphology. Besides demonstrating successful evolution on the specified tasks, the results show that evolved robots make use of the unique body dynamics of soft robots to steer effectively with only a limited controller and actuator design.

Contents

1

Introduction

Traditional, or "hard" robotics deals largely with rigid materials, often actuated by electric motors. In opposition to this, the emerging field of *soft robotics* aims to create robots from soft, flexible and elastic materials. Soft robots are potentially cheaper and lighter than traditional robots, and could interact more safely with people, and with soft objects or terrain[1][2][3]. Besides this, an important concern in soft robotics is the shifting of computational tasks from the controller to the morphology, or to an interaction between controller and morphology. A concrete example is the difference between a hard robotic hand, requiring a complicated design with many joints, sensors, actuators and a complex and fine-tuned controller, and a soft robotic gripper that takes advantage of the compliance of elastomeric materials to automatically wrap around the gripped object[2][4]. The shifted balance between morphology and control makes soft robots a very interesting target for evolutionary robotics.

The complex behaviour of soft materials and robots means hand-designing robots based on human intuition is difficult[5], and existing design techniques for rigid robots don't translate directly to soft robots[6]. This motivates the application of evolutionary robotics to soft robots from an engineering perspective.

The work of Cheney et al., where soft robots are evolved using the voxel-based VoxCAD simulator[7], demonstrates that soft robots can be evolved to perform comparatively complex tasks such as gait learning with only an elementary control system, in this case a fixed periodic expansion and contraction of certain voxels. A fuller overview of this line of voxel-based evolutionary soft robotics research is given in section 2. The present work builds on this work, using similar voxel-based encodings and evaluating robots in the VoxCAD simulator. A limitation of previous evolutionary soft robotics research is that the actuation method does not correspond to that used in (physical) soft robotics practice, where so-called PneuNets, networks of pneumatic actuators, are used[2]. Therefore, I extend the simulator with a more realistic model of pneumatic actuation, to close the reality gap between voxel-based simulations and the real-world dynamics of soft pneumatic actuation. My research question is:

- What is the impact of pneumatic actuation and sensing, including proprioception, on the morphology and task performance of evolvable pneumatic soft robots?

Since, as mentioned above, soft robots are in a unique position to shed light on the interaction

between morphology and control in an evolutionary setting, I address the secondary research question:

- What division of tasks emerges between controller and morphology in evolved pneumatic soft robots?

To answer these questions, experiments are designed to simultaneously evolve pneumatic soft robot bodies and their corresponding recurrent neural network controllers. To investigate the effects of pneumatic sensing and proprioception, both open-loop and closed-loop controllers are used. As fitness functions, both gait learning and directed locomotion are used, to see how the answers to the research questions change as the task gets more difficult.

2

Related work

This chapter introduces key concepts that form the related work of the present research. Firstly, an overview of soft robotics is given, with a special focus on autonomous soft robots. Then, existing work in evolutionary soft robotics is reviewed, forming the main background for my thesis. Finally, some issues regarding the simultaneous evolution of brain and body are highlighted.

2.1. Soft robotics

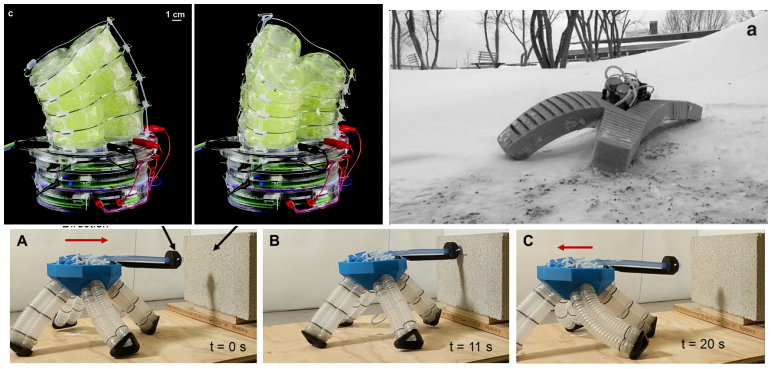


Figure 2.1: Some examples of soft robots: the HASEL actuator[8], an untethered soft robot[9], and an electronics-free walker[10]

Soft robotics is the field of research that aims to create robots from soft, flexible and elastic materials. One motivation for soft robotics is the principle of compliance matching. This is the idea that loads are evenly distributed, and interfacial stress is minimized, when two contacting materials have similar rigidity or softness[1]. It follows that robots which travel over soft terrain should themselves be soft. Also, robots can more safely interact with people and certain soft objects (e.g. delicate fruits and other foods[2]), potentially enabling applications in areas such as agriculture and agriforestry[3].

Cost and simplicity are another advantage of soft robots: some functions which, in a hard robot, would require a very complicated design with many sensors, actuators and computers,

such as a robotic hand, can be solved in a simpler and cheaper way by a soft robot that takes advantage of the compliance of elastomeric materials[2][4].

An early step towards soft robotics was made in the 1950s with the development of McKibben linear actuators, a type of artificial muscle composed of an inflatable bladder constrained inside a braided or woven mesh[2][11]. In the 1980s, Suzumori developed a soft actuator consisting of a flexible cylinder, internally divided into three pneumatic bladders[12]. This formed a prototype of the design pattern of independently inflatable bladders that forms the basis of much of soft robotics today.

Besides the obvious advances in computers and classical robotics, Whitesides[2] identifies the following independent developments that have come together to make soft robotics possible:

- PDMS, an elastomeric silicone material with suitable properties (transparent, sterilizable, biocompatible, easy fabrication)
- Soft lithography and microfluidics, enabling pneumatic control systems
- Digital fabrication, especially for making molds
- Soft composites to create anisotropic motion on actuation

Currently, the softness of robots as a whole is limited by the need to use non-flexible components such as air compressors, circuit boards and sensors. Much of soft robotics research takes place using an external power source, such as an air compressor, and external control systems including valves. Untethering from this base to create autonomous soft robots is still a challenge, largely because of the aforementioned rigidity and weight of the necessary parts[13]. Evidently, soft robotics presents a number of challenges that are unique to the field of robotics, where established solutions from hard robotics are not available, or don't align with the goals of soft robotics. Control systems are one such area.

2.1.1. Materials and fabrication

Soft robots made out of PDMS are usually cast in molds, which can be 3D-printed[2]. Recent advances in 3D printing have made it possible to print soft robots directly. Stereolithography (SLA), selective laser sintering (SLS), fused deposition modeling (FDM), and direct ink writing (DIW) are all applicable to soft materials, although the need for support material may be greater compared to printing hard materials[14]. Of these, direct ink writing currently offers the widest range of materials relevant for soft robotics[14]. Multi-material 3D printing enables the manufacture of functional internal structures, such as pneumatics[15] and varying material stiffness[16].

Besides PDMS, or silicone rubber, materials used in soft robotics include hydrogels, which can be softer and more flexible than PDMS[17], plastic sheet material[17][18], and a variety of smart materials. Smart materials are defined by their capacity to change certain properties (often shape, size or stiffness) in response to a stimulus, such as electric current, heat, light, or magnetism[14]. The 3D printing of smart materials, including the specific way they react to stimuli, is called 4D printing. This enables, for example, the printing of self-folding origami structures in their unfolded state, and smart valves that react to temperature or pH level, acting as both a sensor and actuator[19].

Hard-magnetic soft active materials are a type of smart material created by embedding hard (i.e. permanent) magnetic particles inside a soft polymeric matrix[20]. Wu et al.[20] use an evolutionary algorithm to evolve the density and magnetization direction of the magnetic particles in a voxel-based method, and successfully evolve a quadruped gait actuated by an oscillating magnetic field.

Some soft sensors can also be 3D-printed, paving the way for the manufacturing of complete robots including actuators, sensors, control systems and power supplies through a single process[15].

Another approach is creating soft robots from multiple layers of plastic sheets, which are selectively heat-sealed to create pneumatic structures. This technique is used in e.g. [18] and [21]. The latter takes advantage of the sheet material by keeping the vine-like robot rolled up on a spool.

In 2002, Whitesides and Grzybowski[22] described the potential importance of self-assembly for robotics, where structures can be said to manufacture themselves, instead of relying on external machinery. Recently, self-assembly of 3D structures was achieved in hydrogels by Huang et al.[23]. In contrast to the usual top-down approach to manufacturing, this bottom-up approach forms a potential basis for the physical implementation of developmental soft robotics.

2.1.2. Actuators

An essential part of soft robotics is the use of soft actuators. A common design pattern is the use of inflatable elastic bladders, controlled by a pneumatic network acting as the control system[2]. Using these techniques, soft grippers, octopus-like arms, and walking four-legged robots have been developed[2].

Vacuum can be used as well as positive air pressure, for example by inducing a specific buckling pattern by sucking air out of a network of internal chambers[24]. Ogawa et al.[25] present a modular design system consisting of soft blocks that bend, shear, and shrink isotropically or anisotropically, depending on their internal structure, when a vacuum is applied. Reis[26] gives an overview of the possibilities of buckling as a useful phenomenon, instead of a failure mode. Vacuum can also be used to induce jamming of a soft bag of granular material, freezing its current shape[27]. This principle was used very successfully to create a jamming gripper[28].

Naturally, smart materials are used to create actuators: electroactive polymers support electric activation to create dielectric elastomer actuators (DEAs)[17], shape memory polymers use thermal activation[24], and magnetic or electromagnetic actuators make use of magnetism[24]. HASEL (hydraulically activated self-healing electrostatic) actuators combine the versatility of hydraulic actuators with the performance of DEAs[29][8].

The McKibben muscle already proved the value of combining soft with semi-soft materials, which are flexible but not elastic. This can provide structural strength while still embodying soft robotic principles. Similar possibilities include origami and kirigami (i.e. origami that includes cutting) robots[2][26], and tensegrity[30], where structures are built from beams connected at their endpoints by wires.

2.1.3. Sensors

One of the attractions of soft robots is a reduced reliance on the traditional sensor → controller → actuator feedback loop. In other terms, we might view, for example, a robotic arm that stops bending when it has gripped an object tightly as a combination of sensor and actuator. Nevertheless, work on soft robotic sensors has proved valuable. A buckling soft beam can function as a pressure sensor, by blocking airflow when it is sufficiently compressed [2, fig. 6b]. Electronic bending or stretching sensors can be applied to a soft actuator to provide proprioception. Sensing can also be integrated in the structural material of the robot itself, using semi-conductive polymers that change their conductivity in proportion to their compression or stretching. Conductive hydrogels can be similarly used[17]. Recently, Zou et al.[31] have implemented sensing and closed-loop control in existing pneumatic actuators by measuring the pressure or volume inside the actuators.

2.1.4. Autonomous mobile soft robots

Pneumatic soft robots generally rely for their power on a source of compressed air. Almost all research in soft robotics is performed with robots tethered to an external compressor, power source, and sometimes control system as well. Untethering a soft robot is more difficult than untethering a hard robot, because soft actuators are generally too weak to carry the heavy load of a compressor and batteries[2]. Thus, one path towards untethering soft robots is developing actuators strong enough to carry the required heavy components. This path is taken by Tolley et al.[9], who developed a large (0.65m in length) quadruped soft robot strong enough to carry its own air compressor, batteries and valves. Several untethered soft robots take advantage of the aquatic environment, using buoyancy to support heavy components within a soft body[13]. In 2014, Marchese et al.[32] presented an autonomous soft robotic fish powered by a hydraulic soft actuator in the tail, with the required power and control components contained in the rigid front part of the body. The robot is 34cm long and can maintain a speed of 15cm/s. This robot was developed further by integrating acoustic remote control and vertical movement, for use in underwater exploration[33]. Similarly, Rossi et al.[34] develop a flexible fish-like robot actuated by shape memory alloys, a type of smart material that contracts or expands in response to temperature. This robot attains a cycle time of about 1 fin undulation per second.

A much smaller (9.3cm) soft robotic fish is developed by Li et al.[35]. This robot is propelled by vibrating fins composed of dielectric elastomer actuators (DEAs), powered from an on-board battery.

Ji et al.[36] developed a small (1 gram and 4cm long) insect-like soft robot capable of autonomously following a path. It moves by scooting on three vibrating legs composed of DEAs. A key innovation is the development of low-voltage DEAs operating at less than 500V, making it possible to use smaller and lighter (off-the-shelf) drive electronics and batteries that can be carried by the robot itself.

Several autonomous combustion-driven jumping soft robots have been developed. The jumper presented by Tolley et al.[37] has three pneumatic legs, used for positioning the body towards the desired jumping direction, centered around an explosive actuator. The authors report that the robot can carry enough fuel for 80 jumps, but were not able to achieve this in

practice due to internal damage caused by the jumps.

Untethering a tethered soft robot might mean placing the external hard machinery, to which the robot is tethered, inside the robot body. However, at the boundaries between hard and soft components, stress concentrations occur, which lead to wear and damage of the soft components. Gradual transitions from soft to hard materials can reduce this hazard[13]. This approach is used in the combustion-powered jumping robot by Bartlett et al.[38], which can perform dozens of subsequent jumps without damaging itself.

As solutions that reduce the need for any hard components at all would be more in line with the philosophy and ultimate aims of soft robotics, Rich et al.[13] stress the need for systems-level integration, where the fabrication of actuators and sensors is combined and functionality is moved away from hard components and implemented in soft matter. Polygerinos et al.[4] similarly propose a future development for soft robotics marked by a further integration of all the functions into a "monolithically integrated organism", where the body of the robot itself is actuator, sensor, computer and power source all at once.

Wehner et al.[39] present a proof-of-concept of an autonomous fully soft robot. It uses catalytic decomposition of an on-board liquid fuel to generate pneumatic power, and microfluidic logic for a simple oscillating controller. While limited in its movements, this robot demonstrates the use of chemical and microfluidic techniques to completely remove the need for hard electronic parts.

A more advanced (though somewhat less soft) autonomous soft robot is the electronics-free pneumatic walking robot described by Drotman et al.[10]. Four legs with three DoF each are controlled by a pneumatic ring oscillator circuit to exhibit a walking gait. Input from a pneumatic touch sensor is used to reverse the walking direction. The circuit controlling the gait and responding to the sensor input is completely composed of bistable soft valves, exploiting the non-linear snap-through dynamics of elastomeric materials.

2.1.5. Morphological computation/physical reservoir computing

Morphological computation is the idea that organisms perform some computational functions with their bodies, as opposed to their nervous systems. Originally a concept used in biomechanics to describe natural organisms, there is now interest from the robotics community to make use of it, and soft robots especially are often described as performing morphological computation.[40][41]

Müller and Hoffman[40] support a narrow interpretation of morphological computation, arguing that some common claims of morphological computation are better understood as "morphology facilitating control" or "morphology facilitating perception"[40]. The one phenomenon they do accept as morphological computation proper is physical reservoir computing (PRC). Reservoir computing denotes a particularly unstructured, recurrent neural network with non-linear activation functions of the neurons. Instead of virtual neurons, such a reservoir can also be formed by the body of a robot. The authors give the following examples: a network of masses and springs, with four outputs connected to legs, to compute a gait; and a tensegrity structure consisting of bars and springs which is used to sense terrain properties, and exhibit a stable gait for that terrain. Nakajima et al.[42] explicitly test the computational capability of a soft robotic, octopus-like arm, with promising results. Bhovad

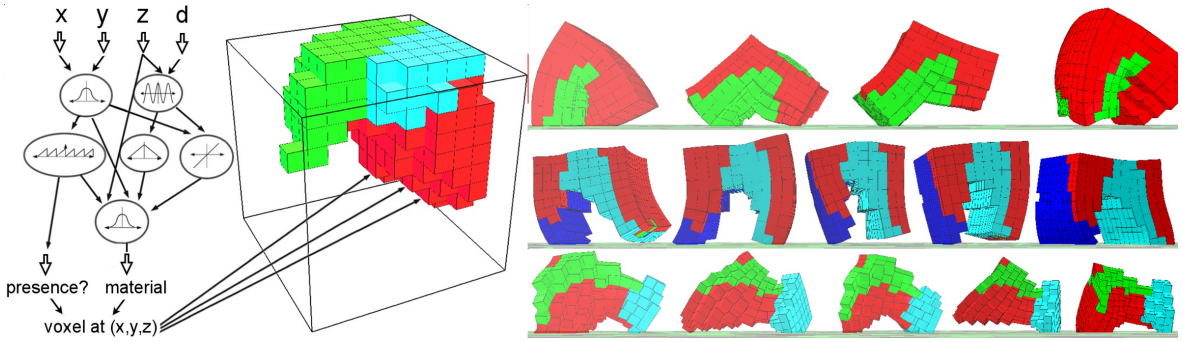


Figure 2.2: Schematic of the CPPN-NEAT encoding, and resulting evolved robots, as used in [7] and related works

and Li[43] show that origami can act as a physical reservoir as well, demonstrating this with a peristaltic, worm-like gait. In general, the following three properties are identified in the literature as essential for a physical computing reservoir: high-dimensionality, i.e. the reservoir has a high-dimensional state space, nonlinearity, and fading (i.e. short-term) memory. The first two properties together provide separation of input signals[40][42][43].

PRC gives another perspective on the difficulty of hand-designing soft-robots. In practice, suitable weights of (neural) reservoir computers are almost universally computed by training, since hand-designing is considered infeasible. Similarly, to optimally utilize the reservoir computing-like properties of a soft robot, automated design (such as via EC) is probably required.

2.2. Evolutionary soft robotics

This section reviews existing work in evolutionary soft robotics.

2.2.1. Voxel-based evolutionary soft robots

Perhaps the largest body of work in evo-soro uses robots composed of active and passive voxels, with the active voxels periodically expanding and contracting as a means of actuation. These robots are usually simulated in VoxCAD[44] or a simulator derived from it.

In [16], robots are encoded using the level-set method, in which a 3D model is generated by applying a threshold to a density field. In this case, the density field is created using a gaussian mixture model.

As demonstrated by Hiller et al.[16], using multi-material 3D printing a soft robot can be manufactured in one pass. When placed in a low-pressure environment, some materials expand more than others. Using this principle, the robots are actuated by placing them inside a pressure chamber that periodically cycles the air pressure. Thus, these robots are not fully autonomous, but have nonetheless proven useful for experiments in locomotion.

In 2014, Cheney et al. investigated soft robot evolution based on a generative encoding called compositional pattern-producing network (CPPN)[7]. CPPNs were already used to provide the controllers in Sims' classic work in evolutionary robotics[45], and were later used as a generative encoding for the network weights in the HyperNEAT neuroevolution

algorithm[46]. Here, they are used to specify a robot from a cube of voxels, where for every coordinate the CPPN is queried to determine the absense or presence, and if present, the material stiffness of the material of that voxel. A CPPN is a neural network where some of the nodes consist of mathematical functions such as sine or gaussian[7]. Compared to a direct encoding, this method produces rather natural-looking, contiguous sections of material, and the evolved gaits are more effective. The robots are also actuated inside a cycling pressure chamber. Cheney et al.[47] expand on this work by using a task tailored to soft robotics, escaping from a box with narrow openings. Cheney et al.[48] present similar work, with the difference that actuation occurs by (simulated) electrical waves through the robot body.

Multi-material, voxel-based robots evolved with CPPN-NEAT are also used by Methenitis et al.[49], where novelty search with fitness-elitism is found to outperform either pure novelty or fitness-based evolution.

Following Joachimczak et al.[50], Lipson et al.[51] enable more complex actuation of voxel-based soft robots by giving individual cells a phase offset.

Corucci et al.[52] simulate voxel-based robots for both walking and swimming. Ogawa [53] also investigates both land- and water-based robots. These robots include sensors in the voxels, and the morphology and control are jointly evolved for multiple tasks simultaneously (i.e. MOO: multi-objective optimization), such as goal finding and locomotion. This results in more complex morphologies than robots that are merely optimized for linear locomotion speed. De Carlo et al.[54] similarly find that in a two-task MOO scenario (locomotion and rotation) where a rigid robot morphology and controller are evolved together, the resulting morphology differs from those resulting from optimization for either task in isolation.

Kriegman et al.[55] in 2020 developed a scalable approach to manufacture voxel-based soft robots. The system is based on individually actuated silicone modules, with a single module per voxel. Although the general shape change of the robots is accurately transfered to the real world, the locomotion itself is not, with the real robots often moving in a different direction than the simulated robots. The authors hypothesize this is caused by a mismatch in friction between the simulation and the real world, although friction parameters that made the simulation agree with reality could not be found[55].

Sui et al.[56] also present a modular physical implementation of voxel-based pneumatic robots, simulated in VoxCAD. In this case, modular units of $5 \times 5 \times 5$ voxels with hard edges and soft faces are used, and locomotion of various morphologies transfers correctly from simulation to reality.

The Evolution Gym, proposed by Bhatia et al.[57] in 2021, is a benchmark of tasks for evolutionary soft robotics. It uses a two-dimensional voxel environment, which limits the applicability to real-world robots, but greatly improves simulation speed. The tasks are graded according to complexity, ranging from simple locomotion to climbing and carrying objects. As a baseline the authors apply CPPN-NEAT to evolve soft robots (bodies and controllers), which is successful on some, but not all, of the tasks. Two-dimensional voxel-based soft robots are also used by Medvet et al.[58] as a platform to investigate diversity when optimizing both the body and the controller (a cyclical per-voxel expansion, similar to that used in the 3D robots above). Ferigo et al.[59] explore evolvability in the context of MAP-Elites using similar 2D robots.

2.2.2. Other work

Besides the popular voxel-based method, other work includes Rieffel et al.[5], using a developmental encoding based on an L-system operating on the faces of tetrahedral meshes. The encoding does not include muscles, instead actuation is provided in simulation by varying the material stiffness.

Joachimzak et al.[50] use a biologically inspired developmental model, incorporating cell division and metamorphosis, which is used to switch between different environments. They also use novelty search to good effect.

In [60] Van Diepen et al. describe a spatial grammar for the description of soft robot, primarily intended for human-guided design. The grammar is also used to generate robots optimized for walking using simulated annealing, hinting at the possibility to use it as a genetic description for an evolutionary algorithm as well.

Fitzgerald et al.[61] use an evolutionary algorithm to find the optimal grain shape and size to use in a jamming gripper, for various arget object sizes and shapes.

The recent master thesis by Pienaar[62] evolves soft robots with realistic pneumatic actuators, using the SOFA simulator (see section 2.2.4 below). A key contribution of this work is the use of randomized rough terrains, enabled by the capability of SOFA to use arbitrary meshes for the ground.

2.2.3. Living tissue robots

Kriegman et al.[55] present a method for creating autonomous soft robots out of living tissue. The robots are constructed from two types of stem cells of the African clawed frog (*xenopus laevis*), one of which is periodically contracting heart muscle tissue, providing actuation. These so-called xenobots are evolved for locomotion in liquid using a VoxCAD simulation. Since the constructed robots differ significantly from the simulated counterparts, several techniques were used to deal with the reality gap. Noise is applied to the simulation to encourage more robust designs. Further, a build filter is applied to deal with the growth behaviour of the cells, which tend to close small concavities. After an initial run of (in-vivo) experiments, simulation parameters were updated to better match the target environment.

In later work, the aggregate behaviour of swarms of these robots is investigated[63]. Swarms are evolved to collect debris from the environment. To enable the simulation of large groups of soft robots, a GPU-accelerated version of VoxCAD is used. Using comparable methods, xenobots of specific shapes are found to aggregate loose cells from the environment in shapes similar to themselves, resulting in a form of self-replication as these clumps grow together[64]. In in-vivo experiments, up to four generations of offspring are achieved.

2.2.4. Simulation

This section describes other simulators used in evo-soro, besides VoxCAD, which is treated in section 2.2.1. In general, finite element analysis (FEA) is the most widely used method of simulating soft robots[17].

In 2009, Rieffel et al. [6] described the evolution of control systems for two soft robots using genetic algorithms and a PhysX simulation. The robots are segmented, with a caterpillar-like body plan actuated, respectively, by pneumatic pistons and contracting wires. Of most interest is the second robot, which is simulated as a soft body without manual simplification of the model. As a GPU-accelerated physics engine, PhysX enables faster iterations than CPU-based simulators by a factor of about 10 to 50. The evolved gait consists of a retrograde wave with bipedal stepping at the front segment.

This PhysX simulation was developed further for the work presented by Rieffel et al.[5].

Van Diemen et al.[60] attempted to simulate soft robots using the softbody simulation available in the Bullet Physics Engine, but encountered problems with self-collision and friction. These problems were solved by modeling the robot as a collection of overlapping, rigid spheres connected by springs.

Coevoet et al.[65] present a system for simulating soft robots, based on FEA, which includes realistic modeling of pneumatic activation taking into account pressure and cavity size. Inverse kinematics are supported as well. It is a plugin for the SOFA simulator[1], originally meant for medical simulations.

ChainQueen[66] is another soft robot simulator and supports pneumatics as well. ChainQueen is aimed at machine learning applications, and a GPU-accelerated implementation is available.

Caasenbrood et al.[67] describe a soft robot simulator which accurately models the interaction between elastic materials and pneumatic actuation. The authors demonstrate their simulator with the topological optimization of a PneuNet-like bending actuator.

A recent review by Pinskier et al.[68] highlights the possibility of adapting simulation and optimization methods from the well-established field of topology optimization (TO) to soft robotics. TO deals with the problem of finding an optimal distribution (according to a fitness function) of material in a component subject to certain constraints and loads, modeled using FEA, and optimized with either gradient-based or evolutionary algorithms. TO methods have already been used for optimizing soft grippers. The most important missing parts of current TO methods are the modeling of soft robot-specific actuation forces, i.e. forces which are dynamic and potentially distributed over the entire body, instead of at fixed points, and the complex interactions between robot and environment.

2.2.5. Embodied evolution

As the preceding has shown, most evolutionary soft robotics research happens in simulation, and the transfer from simulation to reality is challenging. Another possibility is evolving physical robots directly.

Khazanov et al.[69] use on-line evolution to create a locomotion controller in a tensegrity robot. Since the robot can reliably move to the starting position after each run, the authors manage to run multiple evaluations without human intervention.

Veenstra et al.[70] evolve a controller for the optimal fin movement of an aquatic robot. The controller is evaluated directly on the physical robot, and evolved using CMA-ES.

[1] https://www.sofa-framework.org/

Evolutionary modular robotics, where robot morphologies are evolved by the interconnection of a number of fixed modules, has been extensively studied[71]. The same possibility exists with soft robotics. Zhang et al.[72] review recent advances in modular soft robotics, including suitable connections that transfer pneumatic or mechanical power specific to soft robots. Unfortunately, in many cases the connections reduce the softness of the overall robot.

2.2.6. Developmental

In nature, organisms grow, change and learn during their lifetimes. Developmental evolutionary robotics is the integration of this concept into evolutionary robotics. Kriegman et al.[73] demonstrate the advantage of development, where a superior rolling gait is evolved that would otherwise be difficult to find, because it is only a narrow and steep region in the solution space. Rolling is difficult to evolve, because in the early stages of evolution rolling robots tend to roll over just once, and then lay flat, incurring a fitness pentalty w.r.t. slowly but steadily shuffling, scooting or walking robots. The developmental evolutionary algorithm overcomes this problem by having robots develop from a walking gait into a rolling gait near the end of the simulated timeframe. In subsequent generations, the proportion of rolling time is gradually increased. In effect, this transforms a cliff in the fitness landscape into a smooth hill.[73]

Like in nature, artificial evolution combined with development or learning can exhibit the Baldwin effect[73]. This is the phenomenon where traits that were previously acquired during the development of an organism become assimilated into the genetic code of subsequent generations. When jointly evolving morphology and control of soft robots, Kriegman et al.[73] find that these interact through the Baldwin effect. However, they find that only traits that make the robot robust to changes in other traits are assimilated in this way. In these experiments, the result is a morphology that is robust to mutations in the controller (the rolling robot mentioned earlier).

Mazzolai et al.[74] survey self-growing soft robots, which take inspiration from e.g. the exploratory behaviour of climbing plants, and the stimulus-seeking behaviour of plants in general (tropism). Embedded 3D printing, where a structure is grown from the tip by an integrated 3d printer, is described in [75]. The robot is able to bend by asymmetric deposition of material, and can burrow into a loose granular soil as well as growing upwards. Although unable to support its own weight, the vine-inspired robot by Coad et al.[21] is in other ways more advanced, being able to correct its course using pneumatic soft actuators placed along the entire length of the robot body. The body is stored on a spool in the base.

2.3. Simultaneous evolution of body and brain

Answering the research questions requires evolving both bodies and brains in tandem. For this problem, two general approaches can be identified. Firstly, in what can be dubbed the learning approach, the evolutionary process is split into an outer loop, which evolves the robot body, and an inner loop, which separately evolves (e.g. using CMA-ES) or learns (e.g. using reinforcement learning) a controller for each body. Alternatively, in the purely evolutionary approach, the controller and body genomes are operated on together in a single evolutionary process. The first approach generally requires many controller evaluations per

individual body, and consequently evolves fewer generations of morphologies in the outer loop for a given computational budget. For example, Miras et al.[76] use 100 evaluations per individual for controller learning, and 30 generations in the outer loop, while Le Goff et al.[77] use 200 evaluations for controller learning and 20 generations in the outer loop. In contrast, Cheney et al.[7], who use a purely evolutionary approach, evolve body and controller simultaneously over 1000 generations. In this research I will use the evolutionary approach, as the larger number of morphology evolution steps fits better with the morphology-focused RQs. Moreover, as Hart and Le Goff[78] phrase it, simultaneous evolution of body and control "has the obvious advantage of allowing evolution to discover for itself the appropriate balance between morphological and neural complexity in response to the particular environment and task under consideration". Given the somewhat blurred lines between morphology and control, this is even more relevant in soft robotics.

Simultaneous evolution of body and brain remains difficult, which can be seen as one of the causes of a stagnation in the complexity of evolved creatures in the years since Sims' pioneering 1994 work[45]. Premature convergence of the morphology w.r.t. the controller has been identified as a significant contribution to this problem[51]. Mertan and Cheney[79] have recently reiterated this diagnosis, while demonstrating the existence of high-performing solutions that are found when optimizing only the morphology, yet are out of reach of a simultaneous brain-body evolution process.

Among the attempts to resolve this issue, pleiotropic encodings are especially notable within evolutionary soft robotics. Pleiotropy is the dependence of separate phenotypic traits on a single gene. The idea is that when controller and body features are meaningfully encoded in the same genes, a mutation can affect both features without causing a mismatch between body and control. It should be noted that many of the works cited in section 2.2.1 have the potential to exhibit pleiotropy, since the morphology and the choice of cyclically actuated voxels are encoded in the same CPPN. This comes, however, with the caveat that these controllers are very simple and localized. Marzougui et al.[80] introduce a mechanism to enforce pleiotropy by setting a lower bound on the cosine similarity of CPPN connections influencing respectively the body and the controller of voxel-based soft robots. They find that this approach outperforms the non-pleiotropic and normal (non-enforced) pleiotropic encodings. A pleiotropic encoding for a more complex controller is presented in [81], where a single CPPN encodes the morphology, sensor and actuator placement, and weights of a recurrent neural network controller for a modular, non-soft robot.

2.3.1. Controller types

A number of different controller types are commonly used in evolutionary robotics experiments, whether using the learning approach or the pure evolution approach. For modular robots, a CPG (central pattern generator) network is often used, see for example[76]. CPG units are neural oscillators that can be coupled to influence each other. The layout of the units might follow the body plan of the robots, as in [82] and [83]. In [84], CPG units are used to provide sine and cosine inputs to a layer of RBF (radial basis function) neurons to construct an open-loop base controller for a legged robot, which is part of a larger modular controller.

Le Goff et al.[77] use an Elman network to provide the controllers for modular wheeled robots. An Elman network is a recurrent neural network with a fully connected feed-forward hidden

layer, and a recurrent layer of context units. The context layer allows the network to maintain state and thus use past sensor information. In [85] an Elman network is used for both a wheeled and a legged robot navigation task, though the authors conclude that a CPG network would have been better for the legged robot, because it exhibits cyclic patterns naturally.

3

Methods and Experimental Setup

This chapter describes the robot evolution system, consisting of the following parts: a specification of the robot bodies, and the method for simulating these (section 3.1); a recurrent neural network providing the robot controllers (section 3.2); the CPPN-NEAT evolutionary algorithm, including the CPPN specification (section 3.3); and a method to evolve the controllers together with the bodies (section 3.3.2). Finally, section 3.4 discusses the experimental setup.

3.1. Robot morphologies

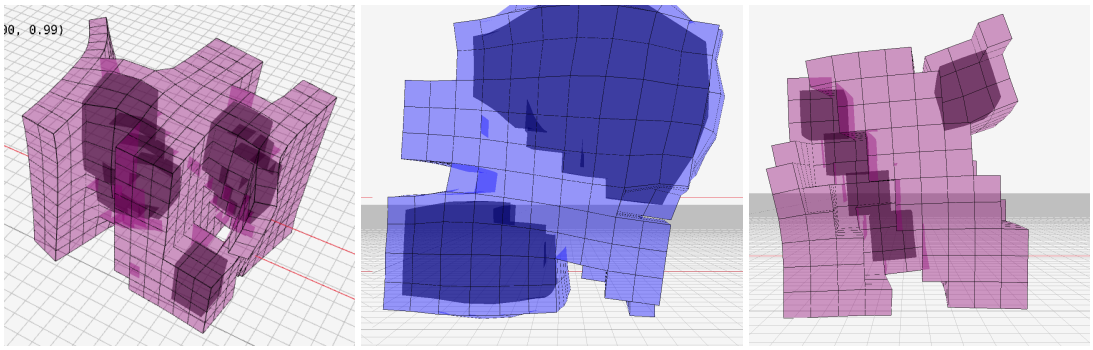


Figure 3.1: Some examples of robot bodies as specified in this section. The colour of the structural voxels ranges from blue (hardest) to red (softest). The robots are rendered transparently, showing the pneumatic chambers on the inside as dark areas.

3.1.1. Robot body properties

The robots used in this research fit into a 10x10x10 lattice of voxels[1]. The material density is set at $1\,000\,\mathrm{kg\,m^{-3}}$. Each voxel is $1\,\mathrm{cm}^3$ meaning robots all fit into a cube 10cm on a side. Each voxel has a stiffness specified in a range between $1 \cdot 10^4$ and $1 \cdot 10^5$ MPa.

[1] robots larger than this would be too time-consuming to simulate

3.1.2. Actuation

Pneumatic actuators are defined as contiguous cavities inside the voxel structure, surrounded by material voxels forming the pneumatic chamber walls. These chambers are connected through separately controllable valves to a positive air pressure source of 1.2 bar, and to the ambient air pressure at 1.0 bar. Thus the controller can direct compressed air into and out of the chamber, providing actuation by blowing up and deflating a section of the robot, as well as possibly keeping a chamber in an inflated or deflated state.[2]

The pneumatics model keeps track of both the air pressure and quantity of air inside the chambers, meaning mechanical forces can be transported from one part of an actuator to another. When the surrounding voxels on one side are compressed, this results in a higher air pressure and thus an expansion at another part of the chamber – something which is not possible when modeling pneumatics using pressure only.

3.1.3. Sensing

The current air pressure inside a chamber can be made available to the controller as a sensor input, thus a single pneumatic chamber can function as both a sensor and an actuator at the same time. This can be used to sense the internal state of the robot, i.e. proprioception, for example by detecting whether a chamber has been inflated up to a specific point; and it can be used for sensing of the external world, such as when contact between the robot and ground causes the air in a chamber to be compressed, resulting in a higher pressure sensor reading. Usage as a sensor will be most effective when both the inlet and outlet valves are closed, though in theory, since the transmission of air through the valves is not instantaneous, a pneumatic chamber could generate useful sensory data even while it is being actuated, e.g. from short-lived spikes in the air pressure.

3.1.4. Simulation

VoxCAD models soft robots as a network of regularly spaced masses (the voxels) which are connected by beams. The beams are modeled as a linear Bernoulli-Euler beam model with both translational and rotational stiffness. For the implementation details of VoxCAD, see [44].

Pneumatics model
Each timestep, the volume of each pneumatic chamber is computed. Combined with the quantity of air the pressure is computed using the ideal gas law:

$$p = n/V * R \tag{3.1}$$

where p is the pressure, n the quantity of air, V the volume, and R a constant representing both the ideal gas constant and a fixed temperature. The resulting pressure is divided over the surface area of all relevant voxels. VoxCAD allows the application of arbitrary force vectors per voxel, which is used to apply the resultant pneumatic force to the walls of the chamber.

[2]The machinery for generating pressure and controlling the valves is abstracted away in this work, so the simulated robots don't have to carry e.g. a compressor or compressed air cylinder.

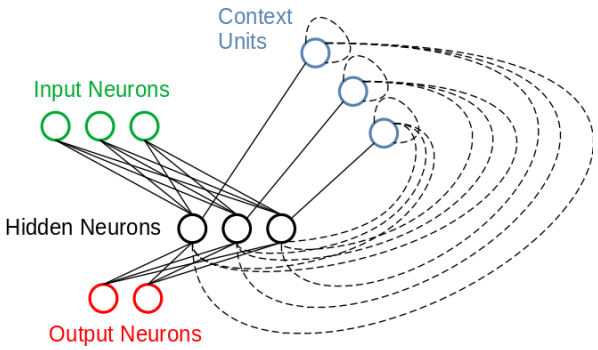


Figure 3.2: Schematic of the controller network. Adapted from [77]

Each step, the quantities of air are transferred into and out of the chamber using the following transfer equation, representing a valve which can be opened or closed in a continuous range:

$$\text{qty} = \text{opening} * \text{conductivity} * (p_1 - p_2) * \text{timestep} \tag{3.2}$$

Where p_1 is the pressure of the pneumatic chamber, and p_2 is either the ambient pressure of 1.0 bar, or the actuation pressure of 1.2 bar, depending the valve in use, *opening* is the control signal, a continuous value between 0 and 1, and *conductivity* is a scaling factor empirically set at 0.025, for a physically plausible behaviour that is not too sudden at fast control changes.

3.2. Robot Controllers

The controller outputs continuous values between 0 and 1 for each valve, which are used to modify the opening parameter in equation 3.2. The inlet and outlet valves are controlled separately, so the controller always has twice as many outputs as there are actuators. To prevent vibrations and numerical instability, a low-pass filter is applied to the output of the controller.

The controller is an Elman network, which is a network with a single hidden layer and a recurrent layer (see fig. 3.2), both consisting of 4 nodes, with tanh activation functions. Following [77], the standard Elman network is modified to have recurrent self-connections at the context units. To aid the controller in generating cyclic patterns, which are absolutely necessary for a viable robot, the controller gets a sin and cosin of adaptable frequency as inputs. The controller can be optionally configured for closed-loop control, gait learning, or both at the same time. In the closed-loop configuration, additional input nodes are added, one for each pneumatic chamber, which pass in the pressure value from equation 3.1, normalized to the range $(-1, 1)$, at each timestep. The directed locomotion configuration adds a sensor input representing the angle between the target direction and the current heading of the robot. This sensor points at a location on the target line (see section 3.4.2) 2 body lengths in front of the current robot position.

3.3. Evolutionary algorithm

The evolutionary algorithm used here builds on that used in [7] and related VoxCAD experiments; see also sec. 2.2.1 for more details. The robot bodies are evolved using

CPPN-NEAT, i.e. NEAT (Neuro-Evolution of Augmenting Topologies) applied to a CPPN (Compositional Pattern-Producing Network).

NEAT parameters
The NEAT algorithm is run for 500 generations with a population size of 50. Tournament selection is used, with a tournament size of 2. The starting point for the CPPN-NEAT hyperparameters were those used in other experiments evolving soft robots with CPPN-NEAT (e.g. [7], [86]), however these resulted, when applied to the present work, in an ineffective evolutionary process: although the maximum fitness increased reasonably over time, the mean fitness stayed very low. This is attributable to the addition of separate (and more complex) controllers instead of the integration of control in the robot body itself as done in the earlier work. Large mutations in the body disrupt the controllability of the body with the pre-mutation controller, making fitness drop dramatically. The final parameters to be used in the experiments were obtained empirically by lowering the mutation strengths until a reasonable development of mean fitness was observed, ultimately lowering the mutation strengths 20-fold. A complete listing of the NEAT parameters is given in table A.1.

CPPN
A CPPN is similar to a regular artificial neural network, but with nodes taking on different transfer functions such as sine, Gaussian, absolute value, etc..., to encode symmetric and repeating patterns more easily. On instantiation of a robot, this network is queried with an x, y, and z value representing a coordinate in the voxel space of the robot body. The output of the network for each query determines the presence and material type of the voxel at that location[7]. I follow earlier implementations (e.g. [7] and [87]) in providing a voxel's distance from the center as an additional input to the CPPN, to bias it towards a more centralized output.

The CPPN has 4 outputs: `presence`, `stiffness`, `is_pneu`, and `sorting_i`. The `presence` output determines whether a voxel is present at all. If a voxel is present, the `is_pneu` output determines whether this voxel is part of a pneumatic chamber, or a structural voxel. If it is a structural voxel, the `stiffness` output determines the material stiffness (Young's modulus) of this voxel, in a range between $1 \cdot 10^4$ MPa and $1 \cdot 10^5$ MPa.

After the initial evaluation, the CPPN output is post-processed to implement the pneumatics. First any structural voxels floating unconnected inside an area of pneumatic voxels are replaced by pneumatic voxels. Then the outer edge of each contiguous area of pneumatic voxels is filled in with structural voxels, according to the `stiffness` output at those coordinates, to ensure pneumatic chambers are properly walled-in. Without this process, only very rarely would a randomly initialized genotype result in a viable (actuatable) robot. Finally, the pneumatic chambers are sorted according to the value of the `sorting_i` output at their centroids. This makes it possible for a consistent mapping between controller in- and outputs and pneumatic chambers to persist throughout mutation and crossover, even while new chambers emerge or disappear.

3.3.1. Encoding of material stiffness

While prior voxel soft robots (e.g. [7] [52]) used individual CPPN outputs for a small number of different materials, with the largest output deciding the material used, my encoding uses

a single CPPN output which - after binning it to a range of 10 possible values for ease of implementation in the simulator - determines the material stiffness. The rationale is to make the search space smoother, by allowing variation operators to cause small shifts in the stiffness of parts of the morphology. It is also a more parsimonious way to encode a finer-grained range of materials, as opposed to using 10 or more separate CPPN outputs, which would complexify the network considerably.

3.3.2. Controller and body evolution

The controller genome consists of a real-valued vector representing the weights and biases of the controller, as well as the frequency of the sine and cosine inputs, which ranges from 1 to 5 Hz. Thus the genome of an individual consists of two sub-genomes: a CPPN for the body and a real-valued vector for the controller. Parent selection and speciation are done by the NEAT algorithm solely based on the CPPN part of the genome. The controller sub-genomes are attached to the body sub-genomes, such that when NEAT dictates that individuals are crossed over and/or mutated, their controller sub-genomes are crossed over resp. mutated as well. The crossover method for the controller is termed *one-point per-layer crossover*, which means standard one-point crossover is applied separately to each layer of the network. This was found, in preliminary experiments, to give better results than either uniform random or standard 1-point crossover. The mutation operator for the controller is self-adaptive mutation, meaning each gene includes a mutation value which is used as the standard deviation of the Gaussian mutations for that specific gene, and is crossed-over together with the rest of the gene. With self-adaptive mutation, the appropriate mutation strength for the controller genes, and thus the balance between controller and body mutation strength, can emerge from the evolutionary process.

3.4. Experimental setup

The four experiments will be labeled as follows: *gait-open* (gait learning with open-loop controller), *gait-closed* (gait learning with closed-loop controller), *dir-open* (directed locomotion with open-loop controller) and *dir-closed* (directed locomotion with closed-loop controller). Each experiment has 10 repetitions.

3.4.1. Gait learning

In the gait learning experiments, the fitness is given by

$$\mathcal{F} = d_x \tag{3.3}$$

where d_x is the x-coordinate of the centroid of the robot after 10 s of simulated time.

3.4.2. Directed locomotion

The fitness function for the directed locomotion experiments is taken from [82]. It is built up out of multiple components, which are evaluated on five target directions (-40, -20, 0, 20, and 40 degrees from the starting position) for each individual, and then added together. For each of the five target directions and corresponding target locomotion paths (i.e. straight lines in

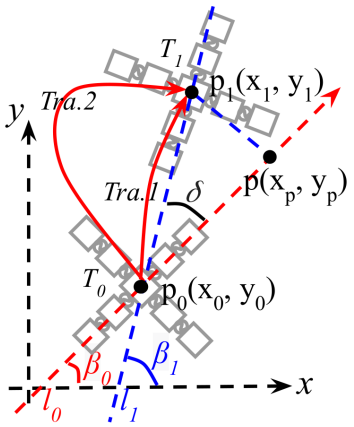


Figure 3.3: Illustration of the directed locomotion fitness function, adapted from [82]. $p_0(x_0, y_0)$ is the starting point of the robot, $p_1(x_1, y_1)$ the final position. l_0 (the red line) is the target line, derived from target direction β_0. $p(x_p, y_p)$ is the projection of the final position on the target line. l_1 is the line through the starting position and the actual final position, and δ is the angle between this line and the target l_0. $Tra.1$ and $Tra.2$ represent possible trajectories the robot might have taken during its evaluation. The length of these trajectories is used, as well as the distance from the target line at 10 regularly spaced sampling points. This process is repeated 5 times for different angles β_0, and the results are summed.

that direction), it rewards the distance moved along the target line, and moving in the right direction in terms of the angle between the target line and the final position. To encourage efficient locomotion, it also penalizes the length of the trajectory, and the distance from the target line at fixed intervals. See fig. 3.3 for an illustration. For each target direction, the component of the fitness function for that direction is given by the following equation:

$$\mathcal{F}_{(\mathcal{D},\mathcal{P},\delta,\mathcal{L})} = \frac{|\mathcal{D}_{(p,p_0)}|}{\mathcal{L}+\varepsilon}\left(\frac{\mathcal{D}_{(p,p_0)}}{\delta_{(\beta_0,\beta_1)}+1} - \mathcal{P}_{(p,p_n)}\right), \tag{3.4}$$

where ε is an infinitesimal constant, and $\delta_{(\beta_0,\beta_1)}$ is the absolute angle between the target line and the line through the start and end positions of the robot in an evaluation run. $\mathcal{L}$ is the length of the trajectory, computed from 10 waypoints sampled at regular intervals during an evaluation run. The component that rewards movement along the target line is

$$\mathcal{D}_{(p,p_0)} = \text{sign}|p - p_0|, \tag{3.5}$$

where p_0 is the starting position of the robot, and p the projection of the final position onto the target line, and where sign = 1 if the robot has moved (roughly) in the correct direction, and sign = -1 if it has moved in the opposite direction. The component that penalizes distance from the target line throughout the trajectory is

$$\mathcal{P}_{(p,p_n)} = \omega \sum_{n=1}^{10} |p_n - \text{proj}_{l_0} p_n|, \tag{3.6}$$

where p_n are the 10 sampled waypoints, and $\text{proj}_{l_0} p_n$ is the projection of each waypoint onto the target line l_0. The ω term is a scalar that determines the strength of this penalty, and is an important tuning parameter. Preliminary experiments showed that a penalty factor of

$\omega = 0.01$, as used in [82], is too small to ensure my evolved robots always follow the target direction. Consequently, the penalty factor is increased to $\omega = 0.1$ in my experiments.

3.5. Implementation notes

The simulator is based on Voxelyze, the backend of VoxCAD, first presented in [44] and currently available at `https://github.com/jonhiller/Voxelyze`. The CPPN-NEAT implementation is based on NEAT-Python[88], available at `https://github.com/CodeReclaimers/neat-python`. The controllers are implemented using Tiny-DNN, available at `https://github.com/tiny-dnn/tiny-dnn`. Finally, the visualization and interactive simulation make use of RayLib, available at `https://www.raylib.com/`. I would hereby like to express my thanks to these authors and developers for making their code publicly available.

4

Results

4.1. Fitness of evolved robots

4.1.1. Gait learning

Figure 4.1 shows the development over generations of the mean and maximum fitness for the gait learning experiments, comparing the open-loop with the closed-loop controllers. The averages of the maximum fitness of each run are roughly equal. In terms of the mean fitness the open-loop controllers outperform the closed-loop controllers, with an average population mean of 40 vs. 26. A t-test shows this difference is statistically significant at $p = 0.012$. Figures 4.3a and 4.3b show the gaits of the top 3 best performing run champions of the *gait-open* and *gait-closed* experiments; the rest of the runs are shown in the appendix.

4.1.2. Directed locomotion

Figure 4.2 shows the development over generations of the mean and maximum fitness for the directed locomotion experiments, comparing the open-loop with the closed-loop controllers. Due to the large variance in the results, the difference between the mean resp. maximum fitnesses are not statistically significant. Figures 4.3c and 4.3d show the gaits, while walking straight ahead, of the top 3 best performing run champions of the *dir-open* and *dir-closed* experiments. Figure 4.4 shows the same robots seen from above, while steering both left and right. Similar figures for the rest of the runs are shown in the appendix.

Videos

A video showing all run champions can be seen at `https://www.youtube.com/watch?v=mJRQac5HiFE`.

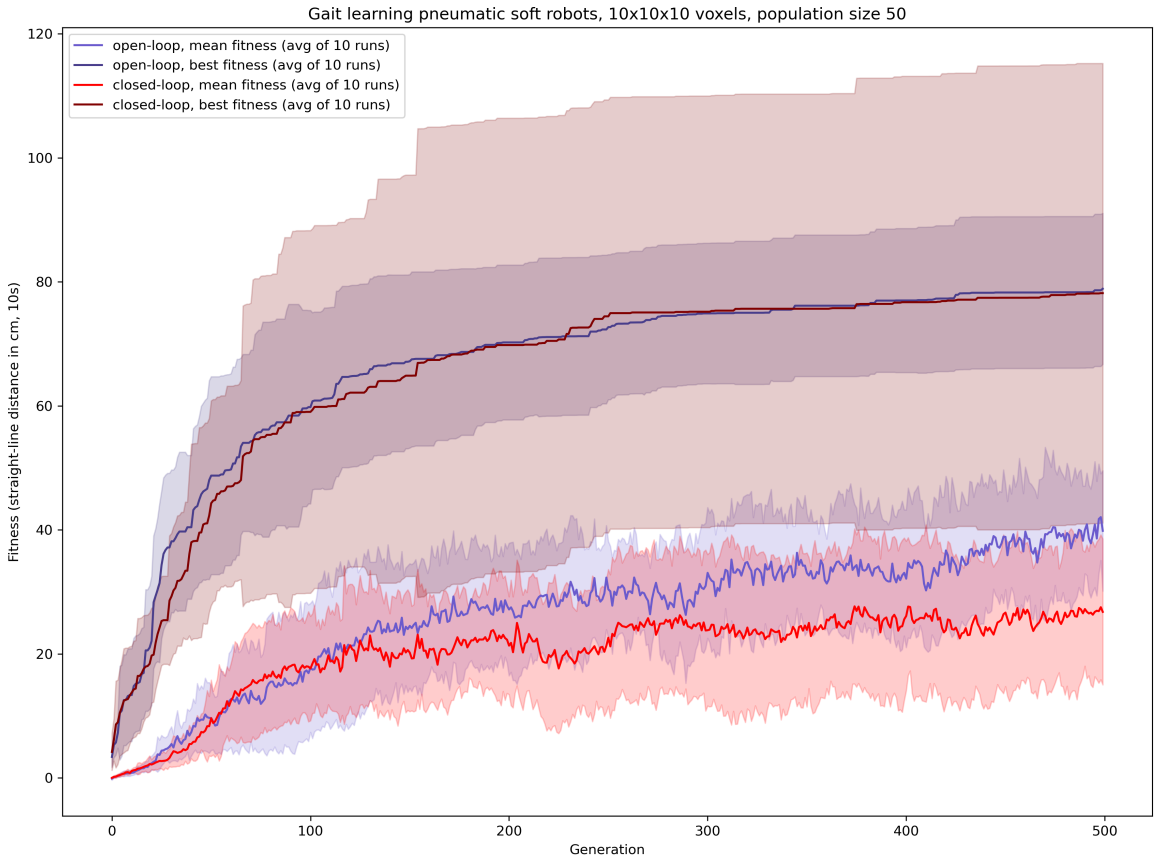


Figure 4.1: Gait learning fitness results, averaged over 10 runs, shaded areas are +/- one standard deviation

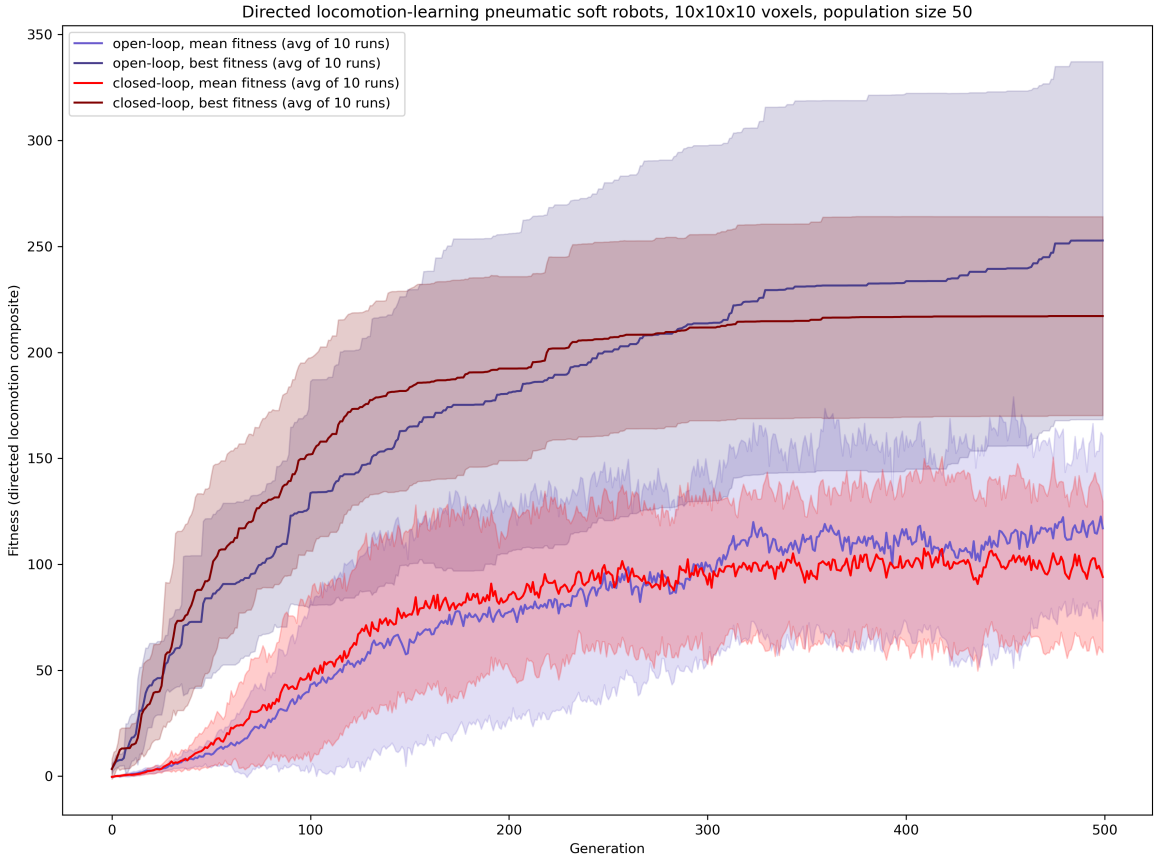
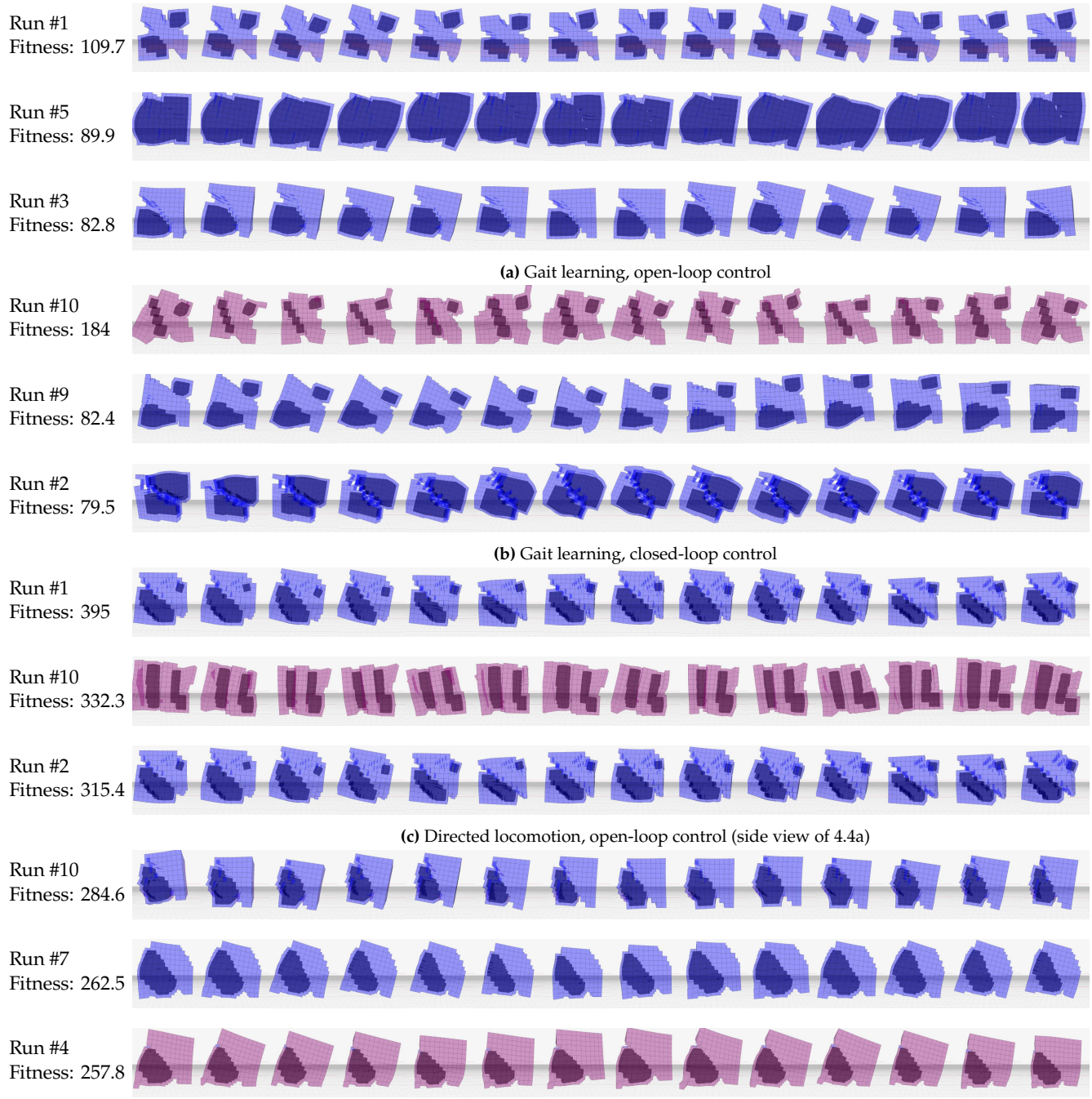


Figure 4.2: Directed locomotion fitness results, averaged over 10 runs, shaded areas are +/- one standard deviation



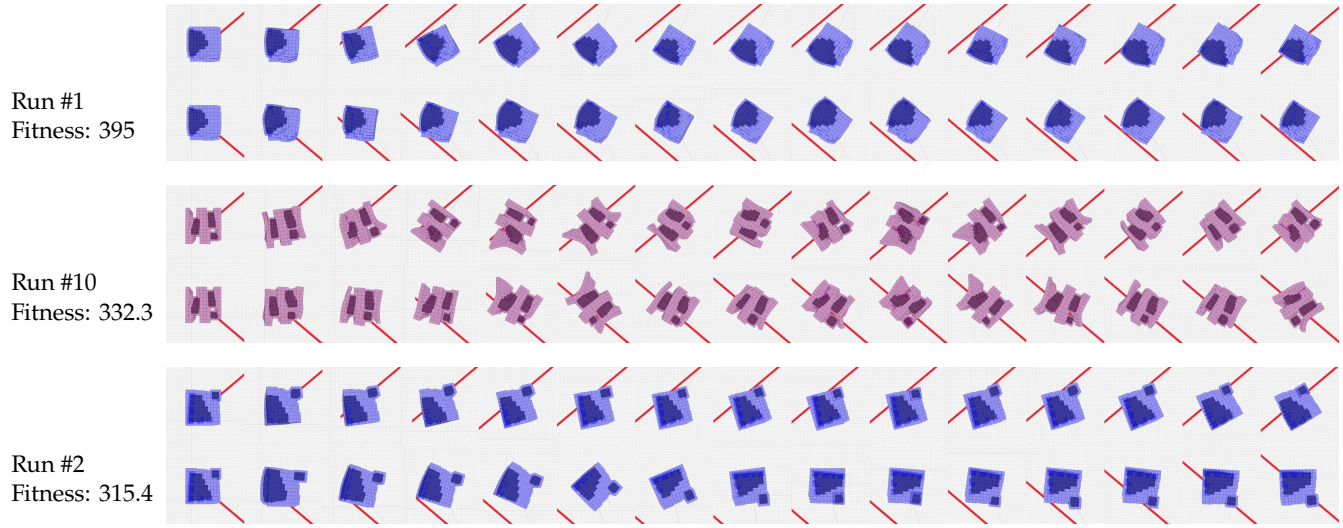
(a) Gait learning, open-loop control

(b) Gait learning, closed-loop control

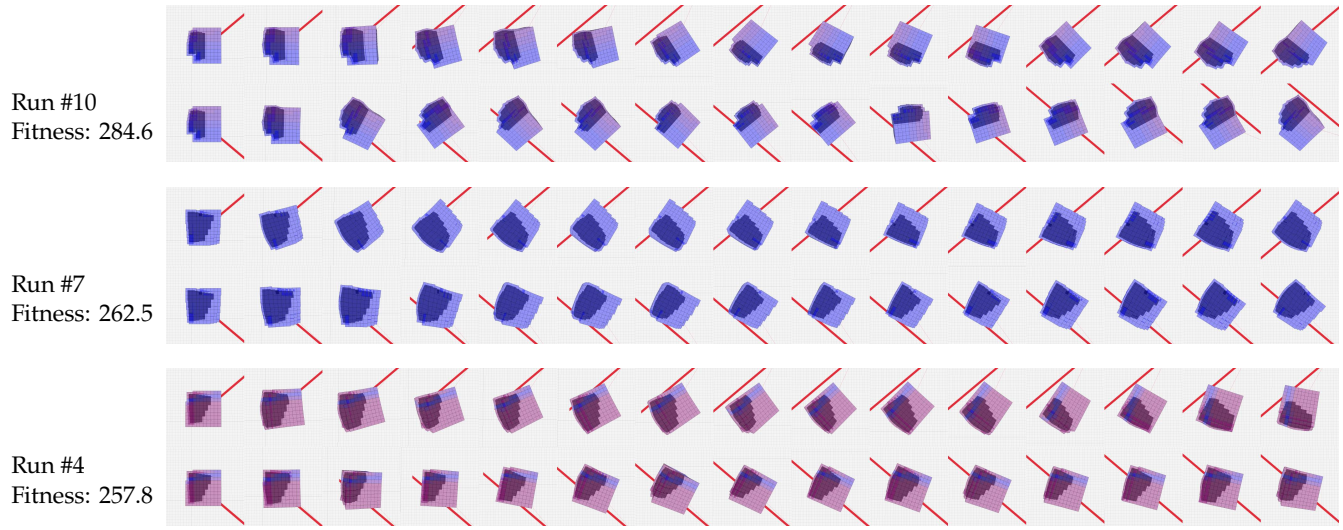
(c) Directed locomotion, open-loop control (side view of 4.4a)

(d) Directed locomotion, closed-loop control (side view of 4.4b)

Figure 4.3: Timelapse images of three *gait-open*, *gait-closed*, *dir-open* and *dir-closed* champions (side view). The robots are rendered transparently; the dark areas inside the robots are the pneumatic chambers. Material colour ranges from blue = hardest to red = softest. Each sequence spans 0.5 seconds. The fitness is calculated according to section 3.4.1 resp. 3.4.2. The run numbers are just labels and have no further meaning. For similar images of all repetitions, see the appendix.



(a) Directed locomotion, open-loop control (top view of fig. 4.3c)



(b) Directed locomotion, closed-loop control (top view of fig. 4.3d)

Figure 4.4: Timelapse of the *dir-open* and *dir-closed* champions seen from above, steering 40 degrees left and right in each pair of images. The red line represents the target direction. The robots are rendered transparently; the dark areas inside the robots are the pneumatic chambers. Material colour ranges from blue = hardest to red = softest. Each sequence spans 10 seconds. The fitness is calculated according to section 3.4.1 resp. 3.4.2. The run numbers are just labels and have no further meaning. For similar images of all repetitions, see the appendix.

5

Analysis and discussion

In general, the results show that the proposed method is successful at both the gait learning and directed locomotion tasks. This chapter reviews the results in detail, closing with some recommendations for future work.

5.1. Qualitative analysis of evolved morphologies and gaits

5.1.1. Gait learning

In most cases the robots exhibit a shuffling gait, alternating between a front and rear "leg" using a single actuator positioned at the rear, and a front leg that bends backward during contact with the ground and flexes back into its neutral position while detached from the ground. Exceptions are *gait-open #5* [1], which has an additional actuator in the front leg and bounces between the two; *gait-closed #2*, which has a single leg and moves by making small jumps, with a second actuator up in the body that might help in generating upward momentum; and *gait-closed #10*, which has multiple actuators in the middle and an efficient two-legged gait.

The emergence of a single large actuator in most robots could be explained by the fact that such an actuator is capable of transforming the largest amount of energy from the pneumatic system into robot motion, given the limited bounding box of the robots. Even when compared to multiple small actuators that sum to the same total actuator volume, the single large actuator is more efficient because it is less impeded by the surrounding structural material. Although a more intricate gait using several smaller actuators might exist in the solution space, the complexification from a single large actuator to multiple small ones would need to overcome the corresponding loss of raw actuation power, and so the single-large-actuator morphology might be a local optimum.

Proprioception
With this potential local optimum in mind, the *gait-closed #10* individual merits special attention: this robot clearly stands above the others in its experiment batch with a fitness of 184cm/10s (while the next-highest fitness is 82cm/10s). It achieves this with a more complex

[1] see appendix A for images and fitness values of specific robots referred to in this chapter

morphology and control scheme of 4 actuators in a staggered activation pattern. Ablation tests reveal that this robot still performs reasonably well with the oscillator inputs disabled, but stops working with the sensor inputs disabled. This shows that this robot works by switching between inflation and deflation of the chambers based on the sensor readings of the air pressure at the actuators, in contrast to most other robots in the batch which rely on the input oscillators. Thus, this is a robot that uses proprioception instead of a fixed oscillator to determine its gait, and outperforms robots evolved with the same experimental parameters that don't use proprioception. Being present in only one of the 10 runs, this was an accidental find, and more research is needed to evolve high-performing robots like this more reliably and confirm that proprioception is indeed the reason for the high performance; see also section 5.3.4.

5.1.2. Directed locomotion

Surprisingly, the evolved morphologies and gaits for directed locomotion are very similar to those in the gait learning experiment, although some robots have the front leg split into a left and right part, presumably aiding in steering. Despite this similarity, these robots are nevertheless successful at directed locomotion. An exception is *dir-open #10*, which has a more complex morphology and controller arrangement than most other robots.

Directed locomotion did not lead to more complex morphologies or the use of multiple actuators in order to support steering. Instead, analysis of the videos reveals that steering happens by changing the phase and/or duty cycle (proportion between high and low output) of the actuation pattern of the single actuator. This results in either the left or the right part of the flexible front leg inhibiting movement more strongly, which causes the steering. Also, some robots perform a whole-body rotation/rolling movement around the vertical axis, and timing the actuation w.r.t. different phases of this motion causes steering.

This development might be attributed, at least partly, to the fact that a robot that performs the same straight-ahead movement for all 5 target directions can still achieve a significant positive fitness. Given the design of the fitness function and experiments, evolution of directionally controllable robots can start out with a dominant selection pressure for simple gait learning, which is later differentiated into steering robots.

5.2. Comparison of open-loop vs. closed-loop control

The experiments comparing open-loop and closed-loop control were designed to detect the potential advantage of soft pneumatic sensing in robots evolved for the specified tasks. The results don't show a quantitative advantage of pneumatic sensing in terms of achieved fitness. This might be attributed to the choice of tasks: effective movement over flat terrain does not necessarily require sensing. However, the fact that open-loop control outperformed closed-loop control in the gait learning experiments hints at the possibility of a deficiency in the evolutionary algorithm: since the open-loop controllers are strictly a subset of the closed-loop controllers, we would not expect them to perform worse when evolved effectively. Therefore, the method of evolving the controller weights might have had too few samples, or suffer from any of the issues commonly identified with brain/body simultaneous evolution

such as premature convergence of the morphology, or a lack of effective crossover between the controllers of different body plans.

5.3. Limitations and future work

5.3.1. Sensing and terrain

Using the current simulator, I was not able to model a physically plausible robot-terrain interaction, in addition to the limitation that the terrain would have to be made out of voxels too. Expanding the simulator with realistic and accurate terrains would open up interesting experiments to test the effect of pneumatic sensing and closed-loop control, which would be a clear advantage on unpredictable or rough terrains.

5.3.2. Morphological computation

I believe that the way the steering behaviour depends on the interaction between the flexible dynamics of the front leg(s) of the robots and subtle timing variations in the controller is a concrete example of "morphology facilitating control", which is theorized in [40] as a more moderate version of the concept of morphological computation. To complete the picture, an experiment designed to bring out the "morphology facilitating perception" concept from the same paper could be a promising avenue for future work.

5.3.3. Directed locomotion

The similarity between robots evolved for gait-learning and directed locomotion has been noted in section 5.1.2. I hypothesized that a robot that simply moves forward is a stepping stone in the fitness landscape on the way to a steering robot. Therefore, it's possible that the morphology is too set in stone by the time diversification into multiple directions happens. To overcome this limitation, a fitness function that penalizes deviation from the trajectory more strongly might find different steering behaviours. Using an in-place rotation objective, or target angles that are further apart, might yield different morphologies and behaviours as well.

5.3.4. Actuator size and robot complexity

We have seen that most robots use a single large actuator, and I conjecture that this actuation type forms a local optimum due to the disproportionately larger actuation power. In future work, this conjecture might be tested by compensating for the loss of power in smaller actuators by increasing the actuation pressure inversely with actuator size, or by limiting the maximum size of actuators, among other possibilities.

However, the robot discussed in section 5.1.1 seems to have overcome this local optimum without such compensatory mechanisms. Instead, this robot is unique in using a closed-loop controller for the timing of its gait, without relying on the fixed oscillators. Therefore we might expect that the use of closed-loop controllers without fixed oscillators could result in more effective and complex evolved robots. A problem with the current setup is that random initialization of a controller without oscillator inputs would seldom result in an actuatable

robot, leading to very poor evolution. This could be addressed in a future experiment by removing the oscillators, while initializing the controllers with a manually constructed closed-loop oscillator for each actuator.

5.3.5. Simultaneous evolution of body and controller

As described in section 3.3, the mutation rates for the morphology were lowered drastically w.r.t. earlier work, to facilitate joint evolution of controller and body. Ideally, larger body mutations would be used without disrupting the controllability. As noted in 3.3.2, joint evolution of body and control is still an open problem. Due to the blurred lines between body and controller in soft robotics, the evolutionary system proposed in this thesis could be a suitable platform to explore this problem. Many different methods for joint optimization of body and control have been proposed (e.g. [89][59][90][86][91][92][93]), and investigating their applicability to the current evolutionary system could be worthwhile.

5.3.6. Controller learning

As mentioned in section 2.3, as an alternative approach to the simultaneous evolution of brain and body, body evolution can be combined with a separate controller learning loop for each body. Section 5.2 mentions the possibility that the controller evolution used in this work might not be effective enough to clearly show the advantages of closed-loop control over open-loop control. Also, as noted in section 3.3, body mutation strengths had to be reduced drastically to avoid disrupting the inherited controllers, which is not a problem when using a controller learning loop. It would thus be interesting to compare both the morphologies and controllers that result from evolution with a learning loop, with the results of the current work.

6

Conclusion

This thesis proposed a framework for evolutionary soft robotics based on an extension to the VoxCAD simulator, with novel contributions including the implementation of a realistic pneumatic actuation and sensing model, a CPPN-based encoding of these actuators+sensors, and the integration of recurrent neural network controllers. In both a gait-learning and directed locomotion setting, this framework was used to investigate the following research questions:

- What is the impact of pneumatic actuation and sensing, including proprioception, on the morphology and task performance of evolvable pneumatic soft robots?
- What division of tasks emerges between controller and morphology in evolved pneumatic soft robots?

The answer to the first RQ can be split into two parts: actuation and sensing.

As regards pneumatic actuation, qualitatively we observe that this results primarily in robots with a single large actuator and a hopping or shuffling gait, of comparable complexity to earlier evolutionary soft robotics work. A small number of robots with multiple actuators and a more complex gait are observed as well. A striking outcome of the directed locomotion experiments is that pneumatic actuation is *not* used to steer directly. Instead, steering behaviour emerges from the interaction between, on the one hand, the bending of unactuated soft appendages, and on the other hand, the timing of the activation of the main actuator. In my view, this is a clear demonstration of the ability of soft robots to perform complex tasks with simple means.

No advantage of sensing of the external world was found either quantitatively in the task performance, nor qualitatively, from manual inspection of the robots to look for sensing behaviour. This could be attributable mainly to the tasks used in this research, which don't necessitate sensing the external world. However, one run champion was found to use pneumatic proprioception to control the timing of its actuators independently of any fixed oscillators, significantly surpassing all other robots in its cohort in terms of fitness, as well as having a greater morphological complexity, making effective use of 4 actuators. This shows the potential importance of self-sensing in evolutionary soft robotics research, which up until now has been used very little.

The second RQ, regarding the division of tasks between controller and morphology, can

be answered separately for either the gait-learning or directed locomotion experiments. In the case of gait learning, we observe that in most run champions the controller provides a cyclic actuation pattern for the actuated rear leg, while the front leg is unactuated and performs the movements required for the shuffling gait independently from the controller, through the soft body dynamics of bending backwards in contact with the ground, and bending forwards while up in the air. In the directed locomotion experiments, we observe a similar division between an actuated rear leg and an unactuated front leg, but with a more complex interplay between the controller and body to facilitate steering. The controller provides a cyclic actuation pattern with different timing characteristics depending on the target direction. The morphology, and especially the flexible front leg, is such that it responds to these variations in the actuation pattern by bending more to the left or to the right.

Besides answering the research questions above, the evolutionary robotics system introduced in this thesis contributes to the state of the art by opening up new avenues of research. The combination of realistic pneumatic actuation and sensing with a suitable encoding and controller makes it possible to evolve soft robots that more closely resemble those designed and manufactured in the physical world. Sensing, and consequently closed-loop control, are currently under-explored in evolutionary robotics research, and especially in evolutionary soft robotics[94]. The proposed system could help to fill this gap.

In conclusion, the proposed framework has been demonstrated to work successfully for evolving realistic soft robots on a variety of tasks. The experimental results underline once more the unique properties of soft robots in blurring the lines between body and brain, and point at the promise of integrating (pneumatic) sensing and proprioception in evolved soft robots. Finally, the gait-learning robots demonstrate effective steering behaviour based on a target direction sensor, allowing interactive control or, potentially, a simple extension to a targeted locomotion behaviour. To the best of my knowledge, this is the first instance of evolved soft robots that are steerable in 3d.

References

[1] C. Majidi, "Soft robotics: A perspective–current trends and prospects for the future," *Soft robotics*, vol. 1, no. 1, pp. 5–11, 2014.

[2] G. M. Whitesides, "Soft robotics," *Angewandte Chemie International Edition*, vol. 57, no. 16, pp. 4258–4273, 2018.

[3] G. Chowdhary, M. Gazzola, G. Krishnan, C. Soman, and S. Lovell, "Soft robotics as an enabling technology for agroforestry practice and research," *Sustainability*, vol. 11, no. 23, p. 6751, 2019.

[4] P. Polygerinos, N. Correll, S. A. Morin, *et al.*, "Soft robotics: Review of fluid-driven intrinsically soft devices; manufacturing, sensing, control, and applications in human-robot interaction," *Advanced Engineering Materials*, vol. 19, no. 12, p. 1 700 016, 2017.

[5] J. Rieffel, D. Knox, S. Smith, and B. Trimmer, "Growing and evolving soft robots," *Artificial life*, vol. 20, no. 1, pp. 143–162, 2014.

[6] J. Rieffel, F. Saunders, S. Nadimpalli, *et al.*, "Evolving soft robotic locomotion in physx," in *Proceedings of the 11th annual conference companion on genetic and evolutionary computation conference: Late breaking papers*, 2009, pp. 2499–2504.

[7] N. Cheney, R. MacCurdy, J. Clune, and H. Lipson, "Unshackling evolution: Evolving soft robots with multiple materials and a powerful generative encoding," *ACM SIGEVOlution*, vol. 7, no. 1, pp. 11–23, 2014.

[8] S. K. Mitchell, X. Wang, E. Acome, *et al.*, "An easy-to-implement toolkit to create versatile and high-performance hasel actuators for untethered soft robots," *Advanced Science*, vol. 6, no. 14, p. 1 900 178, 2019.

[9] M. T. Tolley, R. F. Shepherd, B. Mosadegh, *et al.*, "A resilient, untethered soft robot," *Soft Robotics*, vol. 1, no. 3, pp. 213–223, 2014.

[10] D. Drotman, S. Jadhav, D. Sharp, C. Chan, and M. T. Tolley, "Electronics-free pneumatic circuits for controlling soft-legged robots," *Science Robotics*, vol. 6, no. 51, eaay2627, 2021.

[11] F. Daerden, D. Lefeber, *et al.*, "Pneumatic artificial muscles: Actuators for robotics and automation," *European journal of mechanical and environmental engineering*, vol. 47, no. 1, pp. 11–21, 2002.

[12] K. Suzumori, S. Iikura, and H. Tanaka, "Applying a flexible microactuator to robotic mechanisms," *IEEE Control systems magazine*, vol. 12, no. 1, pp. 21–27, 1992.

[13] S. I. Rich, R. J. Wood, and C. Majidi, "Untethered soft robotics," *Nature Electronics*, vol. 1, no. 2, pp. 102–112, 2018.

[14] J. Z. Gul, M. Sajid, M. M. Rehman, *et al.*, "3d printing for soft robotics–a review," *Science and technology of advanced materials*, vol. 19, no. 1, pp. 243–262, 2018.

[15] T. Wallin, J. Pikul, and R. Shepherd, "3d printing of soft robotic systems," *Nature Reviews Materials*, vol. 3, no. 6, pp. 84–100, 2018.

[16] J. Hiller and H. Lipson, "Automatic design and manufacture of soft robots," *IEEE Transactions on Robotics*, vol. 28, no. 2, pp. 457–466, 2011.

[17] J. Walker, T. Zidek, C. Harbel, *et al.*, "Soft robotics: A review of recent developments of pneumatic soft actuators," *Actuators*, vol. 9, no. 1, p. 3, 2020.
[18] R. Niiyama, X. Sun, C. Sung, B. An, D. Rus, and S. Kim, "Pouch motors: Printable soft actuators integrated with computational design," *Soft Robotics*, vol. 2, no. 2, pp. 59–70, 2015.
[19] F. Momeni, X. Liu, J. Ni, *et al.*, "A review of 4d printing," *Materials & design*, vol. 122, pp. 42–79, 2017.
[20] S. Wu, C. M. Hamel, Q. Ze, F. Yang, H. J. Qi, and R. Zhao, "Evolutionary algorithm-guided voxel-encoding printing of functional hard-magnetic soft active materials," *Advanced Intelligent Systems*, vol. 2, no. 8, p. 2 000 060, 2020.
[21] M. M. Coad, L. H. Blumenschein, S. Cutler, *et al.*, "Vine robots: Design, teleoperation, and deployment for navigation and exploration," *IEEE Robotics & Automation Magazine*, vol. 27, no. 3, pp. 120–132, 2019.
[22] G. M. Whitesides and B. Grzybowski, "Self-assembly at all scales," *Science*, vol. 295, no. 5564, pp. 2418–2421, 2002.
[23] C. Huang, D. Quinn, S. Suresh, and K. J. Hsia, "Controlled molecular self-assembly of complex three-dimensional structures in soft materials," *Proceedings of the National Academy of Sciences*, vol. 115, no. 1, pp. 70–74, 2018.
[24] P. Boyraz, G. Runge, and A. Raatz, "An overview of novel actuators for soft robotics," *Actuators*, vol. 7, no. 3, p. 48, 2018.
[25] J. Ogawa, T. Mori, Y. Watanabe, M. Kawakami, M. N. I. Shiblee, and H. Furukawa, "Mori-a: Soft vacuum-actuated module with 3d-printable deformation structure," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 2495–2502, 2022.
[26] P. M. Reis, "A perspective on the revival of structural (in) stability with novel opportunities for function: From buckliphobia to buckliphilia," *Journal of Applied Mechanics*, vol. 82, no. 11, p. 111 001, 2015.
[27] H. Lipson, "Challenges and opportunities for design, simulation, and fabrication of soft robots," *Soft Robotics*, vol. 1, no. 1, pp. 21–27, 2014.
[28] E. Brown, N. Rodenberg, J. Amend, *et al.*, "Universal robotic gripper based on the jamming of granular material," *Proceedings of the National Academy of Sciences*, vol. 107, no. 44, pp. 18 809–18 814, 2010.
[29] E. Acome, S. K. Mitchell, T. Morrissey, *et al.*, "Hydraulically amplified self-healing electrostatic actuators with muscle-like performance," *Science*, vol. 359, no. 6371, pp. 61–65, 2018.
[30] J. Kimber, Z. Ji, A. Petridou, *et al.*, "Low-cost wireless modular soft tensegrity robots," in *2019 2nd IEEE International Conference on Soft Robotics (RoboSoft)*, IEEE, 2019, pp. 88–93.
[31] S. Zou, S. Picella, J. de Vries, V. G. Kortman, A. Sakes, and J. T. Overvelde, "A retrofit sensing strategy for soft fluidic robots," *Nature Communications*, vol. 15, no. 1, p. 539, 2024.
[32] A. D. Marchese, C. D. Onal, and D. Rus, "Autonomous soft robotic fish capable of escape maneuvers using fluidic elastomer actuators," *Soft robotics*, vol. 1, no. 1, pp. 75–87, 2014.
[33] R. K. Katzschmann, J. DelPreto, R. MacCurdy, and D. Rus, "Exploration of underwater life with an acoustically controlled soft robotic fish," *Science Robotics*, vol. 3, no. 16, eaar3449, 2018.

[34] C. Rossi, J. Colorado, W. Coral, and A. Barrientos, "Bending continuous structures with smas: A novel robotic fish design," *Bioinspiration & biomimetics*, vol. 6, no. 4, p. 045 005, 2011.

[35] T. Li, G. Li, Y. Liang, *et al.*, "Fast-moving soft electronic fish," *Science advances*, vol. 3, no. 4, e1602045, 2017.

[36] X. Ji, X. Liu, V. Cacucciolo, *et al.*, "An autonomous untethered fast soft robotic insect driven by low-voltage dielectric elastomer actuators," *Science Robotics*, vol. 4, no. 37, eaaz6451, 2019.

[37] M. T. Tolley, R. F. Shepherd, M. Karpelson, *et al.*, "An untethered jumping soft robot," in *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, IEEE, 2014, pp. 561–566.

[38] N. W. Bartlett, M. T. Tolley, J. T. Overvelde, *et al.*, "A 3d-printed, functionally graded soft robot powered by combustion," *Science*, vol. 349, no. 6244, pp. 161–165, 2015.

[39] M. Wehner, R. L. Truby, D. J. Fitzgerald, *et al.*, "An integrated design and fabrication strategy for entirely soft, autonomous robots," *Nature*, vol. 536, no. 7617, pp. 451–455, 2016.

[40] V. C. Müller and M. Hoffmann, "What is morphological computation? on how the body contributes to cognition and control," *Artificial life*, vol. 23, no. 1, pp. 1–24, 2017.

[41] K. Ghazi-Zahedi, J. Rieffel, S. Schmitt, and H. Hauser, "Recent trends in morphological computation," *Frontiers in Robotics and AI*, vol. 8, p. 708 206, 2021.

[42] K. Nakajima, H. Hauser, T. Li, and R. Pfeifer, "Information processing via physical soft body," *Scientific reports*, vol. 5, no. 1, pp. 1–11, 2015.

[43] P. Bhovad and S. Li, "Physical reservoir computing with origami and its application to robotic crawling," *Scientific Reports*, vol. 11, no. 1, pp. 1–18, 2021.

[44] J. Hiller and H. Lipson, "Dynamic simulation of soft heterogeneous objects," *arXiv preprint arXiv:1212.2845*, 2012.

[45] K. Sims, "Evolving virtual creatures," in *Proceedings of the 21st annual conference on Computer graphics and interactive techniques*, ACM New York, NY, USA, 1994, pp. 15–22.

[46] K. O. Stanley, D. B. D'Ambrosio, and J. Gauci, "A hypercube-based encoding for evolving large-scale neural networks," *Artificial life*, vol. 15, no. 2, pp. 185–212, 2009.

[47] N. Cheney, J. Bongard, and H. Lipson, "Evolving soft robots in tight spaces," in *Proceedings of the 2015 annual conference on Genetic and Evolutionary Computation*, ACM New York, NY, USA, 2015, pp. 935–942.

[48] N. Cheney, J. Clune, and H. Lipson, "Evolved electrophysiological soft robots," in *ALIFE 14: Proceedings of The Fourteenth International Conference on the Synthesis and Simulation of Living Systems*, MIT Press, 2014, pp. 222–229.

[49] G. Methenitis, D. Hennes, D. Izzo, and A. Visser, "Novelty search for soft robotic space exploration," in *Proceedings of the 2015 annual conference on Genetic and Evolutionary Computation*, ACM New York, NY, USA, 2015, pp. 193–200.

[50] M. Joachimczak, R. Suzuki, and T. Arita, "Artificial metamorphosis: Evolutionary design of transforming, soft-bodied robots," *Artificial life*, vol. 22, no. 3, pp. 271–298, 2016.

[51] H. Lipson, V. Sunspiral, J. Bongard, and N. Cheney, "On the difficulty of co-optimizing morphology and control in evolved virtual creatures," in *Artificial Life Conference Proceedings 13*, MIT Press, 2016, pp. 226–233.

[52] F. Corucci, "Evolutionary developmental soft robotics: Towards adaptive and intelligent soft machines following nature's approach to design," in *Soft Robotics: Trends, Applications and Challenges*, Springer, 2017, pp. 111–116.

[53] J. Ogawa, "Evolutionary multi-objective optimization for evolving soft robots in different environments," in *International Conference on Bio-inspired Information and Communication*, Springer, 2019, pp. 112–131.

[54] M. De Carlo, E. Ferrante, J. Ellers, G. Meynen, and A. Eiben, "The impact of different tasks on evolved robot morphologies," in *Proceedings of the Genetic and Evolutionary Computation Conference Companion*, ACM New York, NY, USA, 2021, pp. 91–92.

[55] S. Kriegman, A. M. Nasab, D. Shah, *et al.*, "Scalable sim-to-real transfer of soft robot designs," in *2020 3rd IEEE International Conference on Soft Robotics (RoboSoft)*, IEEE, 2020, pp. 359–366.

[56] X. Sui, H. Cai, D. Bie, Y. Zhang, J. Zhao, and Y. Zhu, "Automatic generation of locomotion patterns for soft modular reconfigurable robots," *Applied Sciences*, vol. 10, no. 1, p. 294, 2020.

[57] J. Bhatia, H. Jackson, Y. Tian, J. Xu, and W. Matusik, "Evolution gym: A large-scale benchmark for evolving soft robots," *Advances in Neural Information Processing Systems*, vol. 34, pp. 2201–2214, 2021.

[58] E. Medvet, A. Bartoli, F. Pigozzi, and M. Rochelli, "Biodiversity in evolved voxel-based soft robots," in *Proceedings of the Genetic and Evolutionary Computation Conference*, ACM New York, NY, USA, 2021, pp. 129–137.

[59] A. Ferigo, L. Soros, E. Medvet, and G. Iacca, "On the entanglement between evolvability and fitness: An experimental study on voxel-based soft robots," in *Artificial Life Conference Proceedings 34*, MIT press, vol. 2022, 2022, p. 15.

[60] M. Van Diepen and K. Shea, "A spatial grammar method for the computational design synthesis of virtual soft locomotion robots," *Journal of Mechanical Design*, vol. 141, no. 10, p. 101 402, 2019.

[61] S. G. Fitzgerald, G. W. Delaney, D. Howard, and F. Maire, "Evolving soft robotic jamming grippers," in *Proceedings of the Genetic and Evolutionary Computation Conference*, ACM New York, NY, USA, 2021, pp. 102–110.

[62] M. Pienaar, "Evolving soft robots with cppn-neat in a randomised domain with realistic fluidic elastomer actuators," M.S. thesis, University of the Witwatersrand, Johannesburg, 2023.

[63] D. Blackiston, E. Lederer, S. Kriegman, S. Garnier, J. Bongard, and M. Levin, "A cellular platform for the development of synthetic living machines," *Science Robotics*, vol. 6, no. 52, eabf1571, 2021.

[64] S. Kriegman, D. Blackiston, M. Levin, and J. Bongard, "Kinematic self-replication in reconfigurable organisms," *Proceedings of the National Academy of Sciences*, vol. 118, no. 49, e2112672118, 2021.

[65] E. Coevoet, T. Morales-Bieze, F. Largilliere, *et al.*, "Software toolkit for modeling, simulation, and control of soft robots," *Advanced Robotics*, vol. 31, no. 22, pp. 1208–1224, 2017.

[66] Y. Hu, J. Liu, A. Spielberg, *et al.*, "Chainqueen: A real-time differentiable physical simulator for soft robotics," in *2019 International conference on robotics and automation (ICRA)*, IEEE, 2019, pp. 6265–6271.

[67] B. Caasenbrood, A. Pogromsky, and H. Nijmeijer, "A computational design framework for pressure-driven soft robots through nonlinear topology optimization," in *2020 3rd IEEE International Conference on Soft Robotics (RoboSoft)*, IEEE, 2020, pp. 633–638.

[68] J. Pinskier and D. Howard, "From bioinspiration to computer generation: Developments in autonomous soft robot design," *Advanced Intelligent Systems*, vol. 4, no. 1, p. 2 100 086, 2022.

[69] M. Khazanov, J. Jocque, and J. Rieffel, "Evolution of locomotion on a physical tensegrity robot," in *Artificial Life Conference Proceedings 14*, MIT Press, 2014, pp. 232–238.

[70] F. Veenstra, J. Jørgensen, and S. Risi, "Evolution of fin undulation on a physical knifefish-inspired soft robot," in *Proceedings of the Genetic and Evolutionary Computation Conference*, ACM New York, NY, USA, 2018, pp. 157–164.

[71] R. J. Alattas, S. Patel, and T. M. Sobh, "Evolutionary modular robotics: Survey and analysis," *Journal of Intelligent & Robotic Systems*, vol. 95, no. 3, pp. 815–828, 2019.

[72] C. Zhang, P. Zhu, Y. Lin, Z. Jiao, and J. Zou, "Modular soft robotics: Modular units, connection mechanisms, and applications," *Advanced Intelligent Systems*, vol. 2, no. 6, p. 1 900 166, 2020.

[73] S. Kriegman, N. Cheney, and J. Bongard, "How morphological development can guide evolution," *Scientific reports*, vol. 8, no. 1, pp. 1–10, 2018.

[74] B. Mazzolai, A. Mondini, E. Del Dottore, and A. Sadeghi, "Self-growing adaptable soft robots," in *Mechanically Responsive Materials for Soft Robotics*. John Wiley & Sons, Ltd, 2020, ch. 15, pp. 363–394.

[75] A. Sadeghi, A. Mondini, and B. Mazzolai, "Toward self-growing soft robots inspired by plant roots and based on additive manufacturing technologies," *Soft robotics*, vol. 4, no. 3, pp. 211–223, 2017.

[76] K. Miras, M. De Carlo, S. Akhatou, and A. Eiben, "Evolving-controllers versus learning-controllers for morphologically evolvable robots," in *Applications of Evolutionary Computation: 23rd European Conference, EvoApplications 2020, Held as Part of EvoStar 2020, Seville, Spain, April 15–17, 2020, Proceedings 23*, Springer, 2020, pp. 86–99.

[77] L. K. Le Goff, E. Buchanan, E. Hart, *et al.*, "Morpho evolution with learning using a controller archive as an inheritance mechanism," *IEEE Transactions on Cognitive and Developmental Systems*, vol. 15, no. 2, pp. 507–517, 2023.

[78] E. Hart and L. K. Le Goff, "Artificial evolution of robot bodies and control: On the interaction between evolution, learning and culture," *Philosophical Transactions of the Royal Society B*, vol. 377, no. 1843, p. 20 210 117, 2022.

[79] A. Mertan and N. Cheney, "Investigating premature convergence in co-optimization of morphology and control in evolved virtual soft robots," in *European Conference on Genetic Programming (Part of EvoStar)*, Springer, 2024, pp. 38–55.

[80] D. Marzougui, M. Biondina, and F. Wyffels, "A comparative analysis on genome pleiotropy for evolved soft robots," in *Proceedings of the Genetic and Evolutionary Computation Conference Companion*, ACM New York, NY, USA, 2022, pp. 136–139.

[81] J. E. Auerbach and J. C. Bongard, "Evolving complete robots with cppn-neat: The utility of recurrent connections," in *Proceedings of the 13th annual conference on Genetic and evolutionary computation*, ACM New York, NY, USA, 2011, pp. 1475–1482.

[82] G. Lan, M. De Carlo, F. van Diggelen, J. M. Tomczak, D. M. Roijers, and A. Eiben, "Learning directed locomotion in modular robots with evolvable morphologies," *Applied Soft Computing*, vol. 111, pp. 1–17, 2021.

[83] M. Jelisavcic, K. Glette, E. Haasdijk, and A. Eiben, "Lamarckian evolution of simulated modular robots," *Frontiers in Robotics and AI*, vol. 6, p. 9, 2019.
[84] M. Thor and P. Manoonpong, "Versatile modular neural locomotion control with fast learning," *Nature Machine Intelligence*, vol. 4, no. 2, pp. 169–179, 2022.
[85] L. K. Le Goff, E. Hart, A. Coninx, and S. Doncieux, "On pros and cons of evolving topologies with novelty search," in *Artificial Life Conference Proceedings 32*, MIT Press, 2020, pp. 423–431.
[86] N. Cheney, J. Bongard, V. SunSpiral, and H. Lipson, "Scalable co-optimization of morphology and control in embodied machines," *Journal of The Royal Society Interface*, vol. 15, no. 143, p. 20 170 937, 2018.
[87] J. Clune and H. Lipson, "Evolving three-dimensional objects with a generative encoding inspired by developmental biology," in *ECAL 2011: The 11th European Conference on Artificial Life*, MIT Press, 2011, pp. 2–12.
[88] A. McIntyre, M. Kallada, C. G. Miguel, C. Feher de Silva, and M. L. Netto, *neat-python*.
[89] M. D. Schmidt and H. Lipson, "Age-fitness pareto optimization," in *Proceedings of the 12th annual conference on Genetic and evolutionary computation*, ACM New York, NY, USA, 2010, pp. 543–544.
[90] E. Zardini, D. Zappetti, D. Zambrano, G. Iacca, and D. Floreano, "Seeking quality diversity in evolutionary co-design of morphology and control of soft tensegrity modular robots," in *Proceedings of the Genetic and Evolutionary Computation Conference*, ACM New York, NY, USA, 2021, pp. 189–197.
[91] F. Tanaka and C. Aranha, "Co-evolving morphology and control of soft robots using a single genome," in *2022 IEEE Symposium Series on Computational Intelligence (SSCI)*, IEEE, 2022, pp. 1235–1242.
[92] F. Veenstra, M. H. Olsen, and K. Glette, "Effects of encodings and quality-diversity on evolving 2d virtual creatures," in *Proceedings of the Genetic and Evolutionary Computation Conference Companion*, ACM New York, NY, USA, 2022, pp. 164–167.
[93] J. Nordmoen, F. Veenstra, K. O. Ellefsen, and K. Glette, "Map-elites enables powerful stepping stones and diversity for modular robotics," *Frontiers in Robotics and AI*, vol. 8, p. 639 173, 2021.
[94] A. Ferigo, G. Iacca, and E. Medvet, "Beyond body shape and brain: Evolving the sensory apparatus of voxel-based soft robots," in *International Conference on the Applications of Evolutionary Computation (Part of EvoStar)*, Springer, 2021, pp. 210–226.

A

Appendix

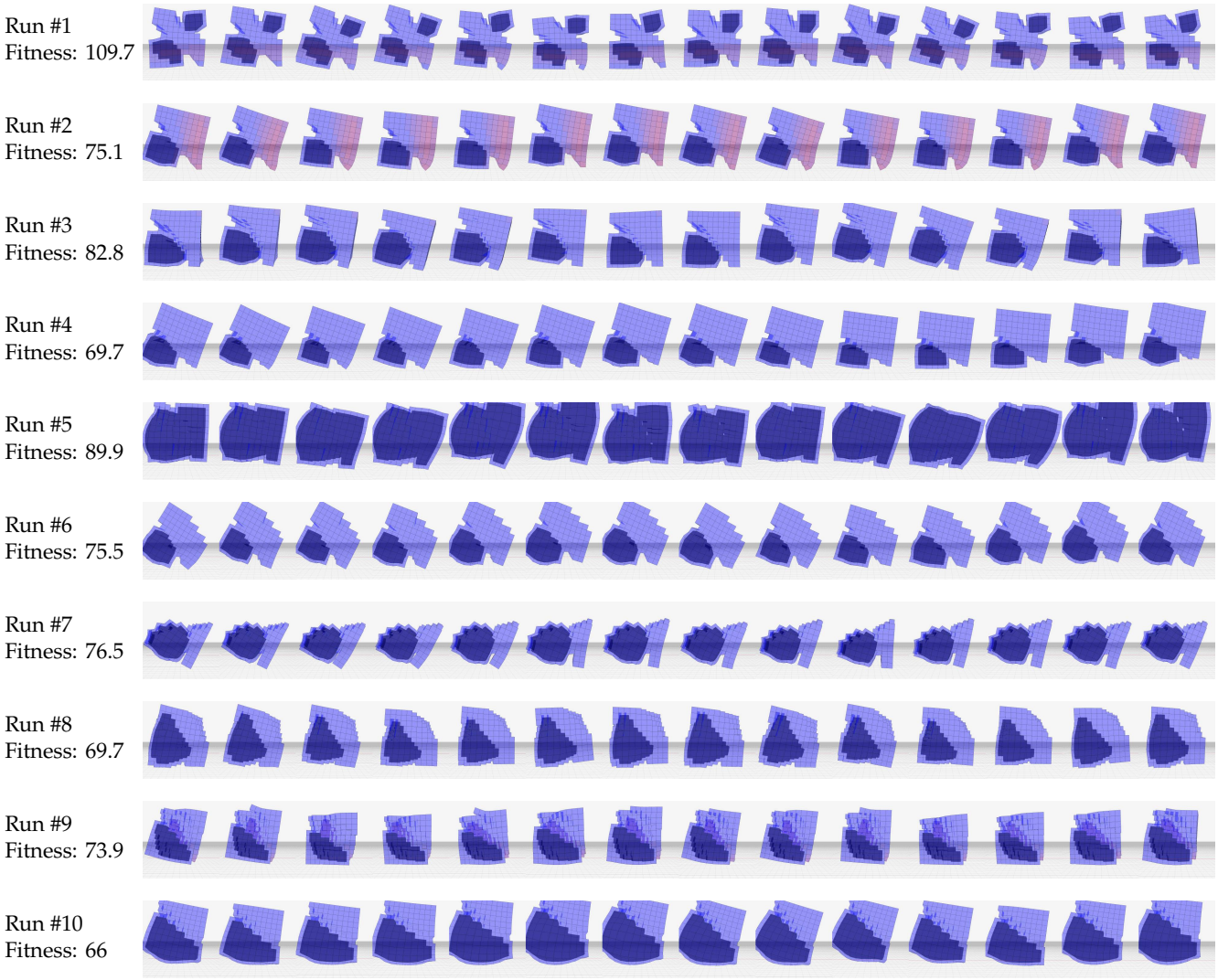


Figure A.1: Timelapse of the *gait-open* champions (side view). The robots are rendered transparently; the dark areas inside the robots are the pneumatic chambers. Material colour ranges from blue = hardest to red = softest. Each sequence spans 0.5 seconds. The fitness is calculated according to section 3.4.1. The run numbers are just labels and have no further meaning.

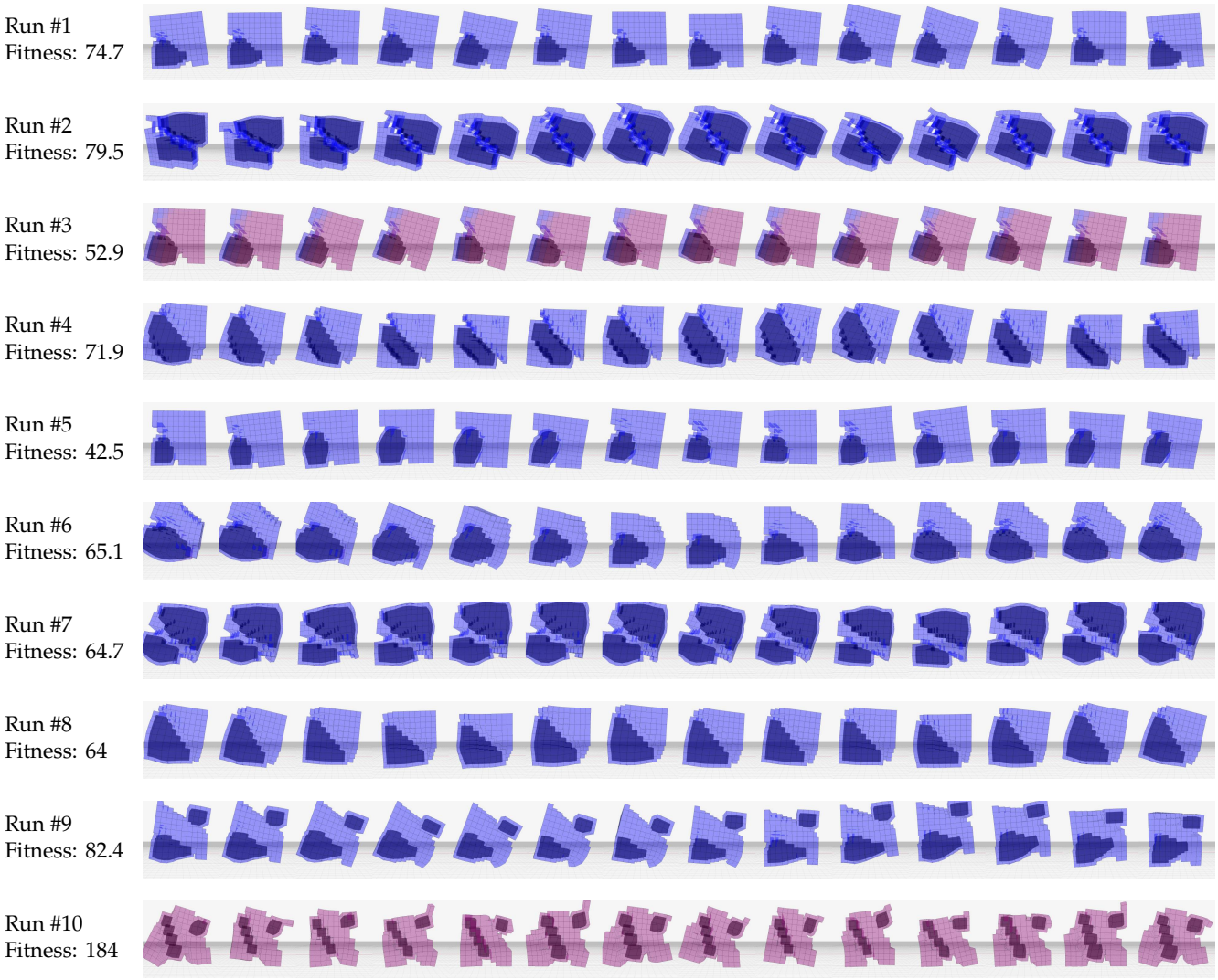


Figure A.2: Timelapse of the *gait-closed* champions (side view). The robots are rendered transparently; the dark areas inside the robots are the pneumatic chambers. Material colour ranges from blue = hardest to red = softest. Each sequence spans 0.5 seconds. The fitness is calculated according to section 3.4.1. The run numbers are just labels and have no further meaning.

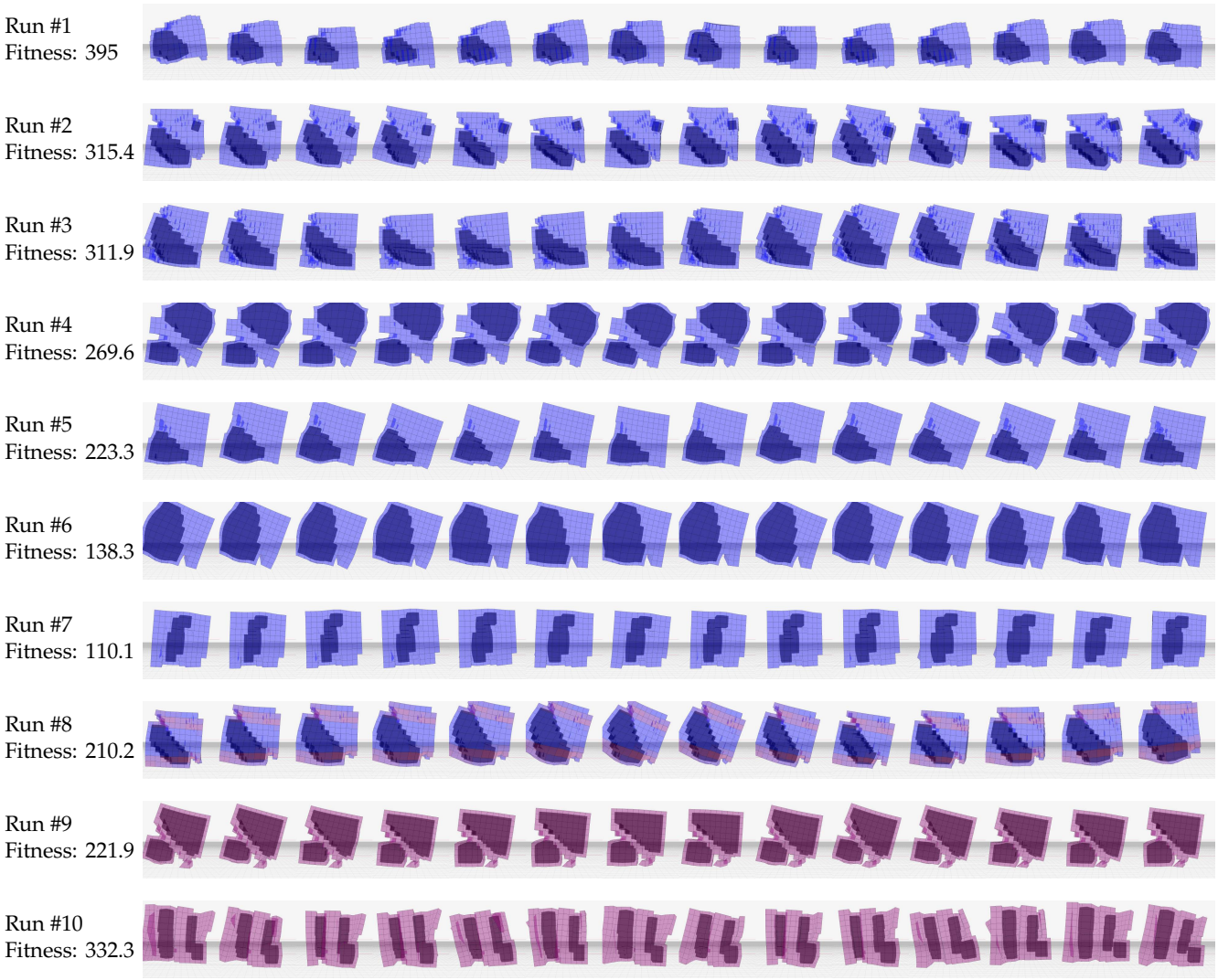


Figure A.3: Timelapse of the *dir-open* champions (side view; corresponds to fig. A.5). The robots are rendered transparently; the dark areas inside the robots are the pneumatic chambers. Material colour ranges from blue = hardest to red = softest. Each sequence spans 0.5 seconds. The fitness is calculated according to section 3.4.2. The run numbers are just labels and have no further meaning.

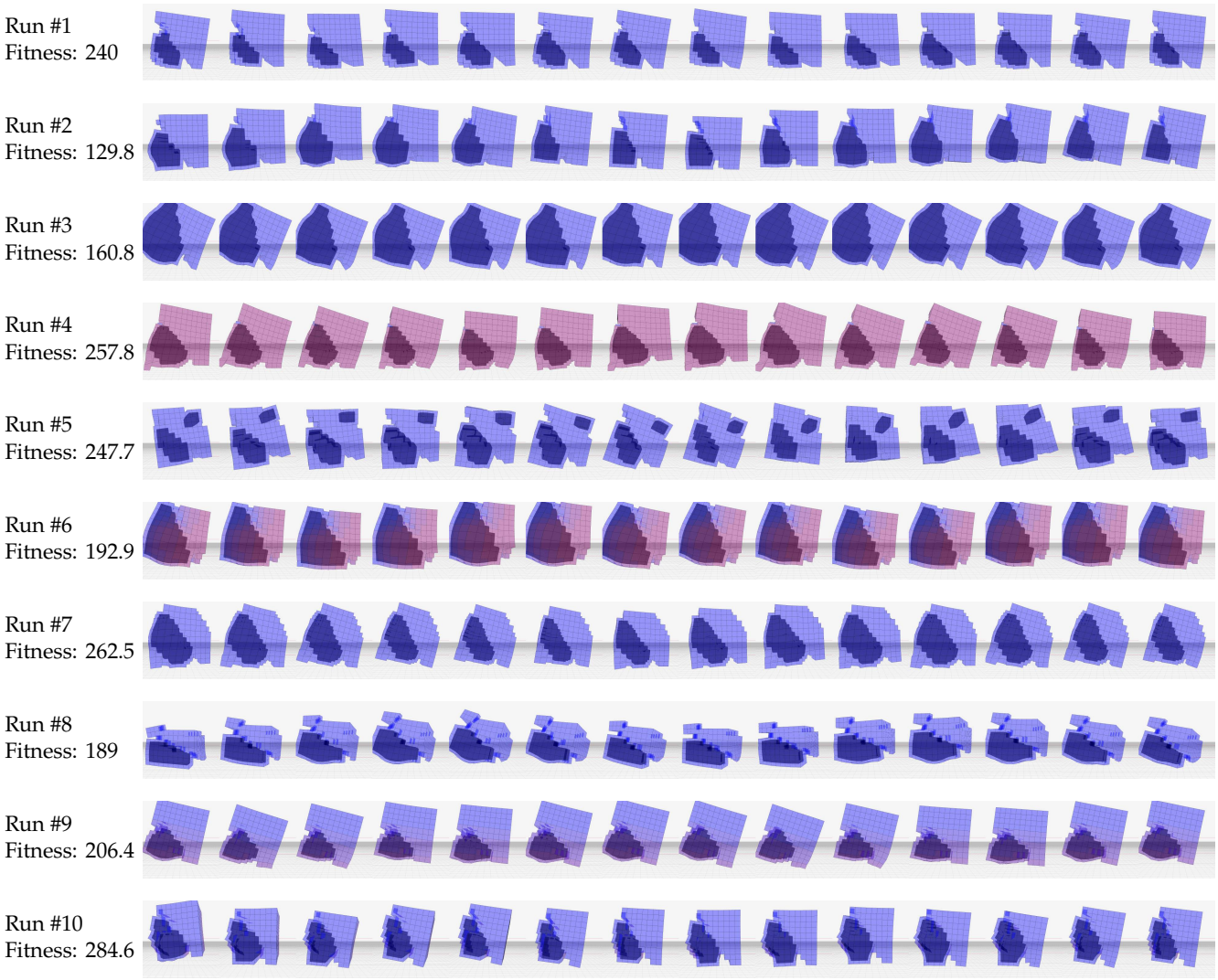


Figure A.4: Timelapse of the *dir-closed* champions (side view; corresponds to fig. A.6). The robots are rendered transparently; the dark areas inside the robots are the pneumatic chambers. Material colour ranges from blue = hardest to red = softest. Each sequence spans 0.5 seconds. The fitness is calculated according to section 3.4.2. The run numbers are just labels and have no further meaning.

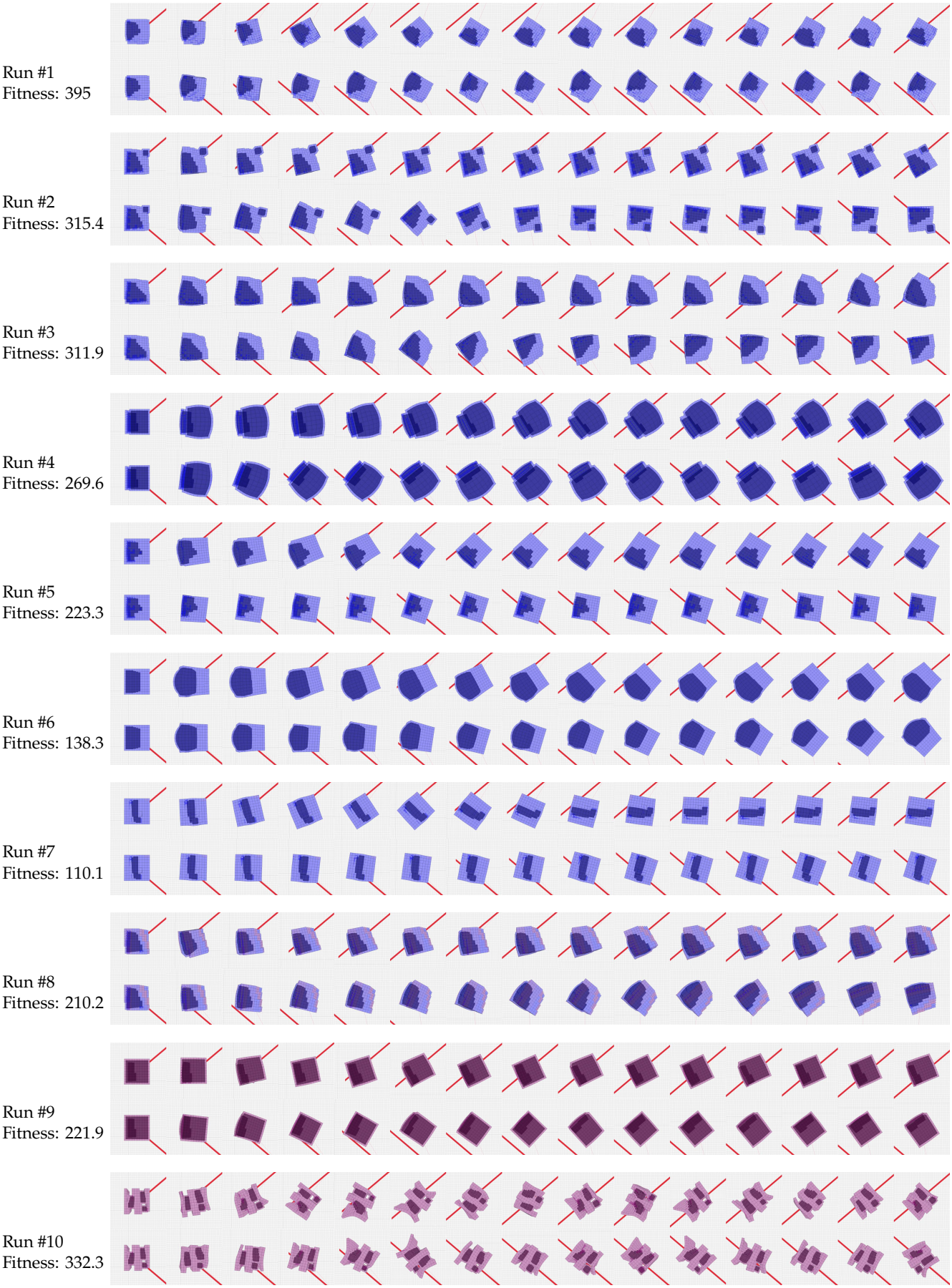


Figure A.5: Timelapse of the *dir-open* champions seen from above (corresponds to fig. A.3), stearing 40 degrees left and right in each pair of images. The red line represents the target direction. The robots are rendered transparently; the dark areas inside the robots are the pneumatic chambers. Material colour ranges from blue = hardest to red = softest. Each sequence spans 10 seconds. The fitness is calculated according to section 3.4.2. The run numbers are just labels and have no further meaning.

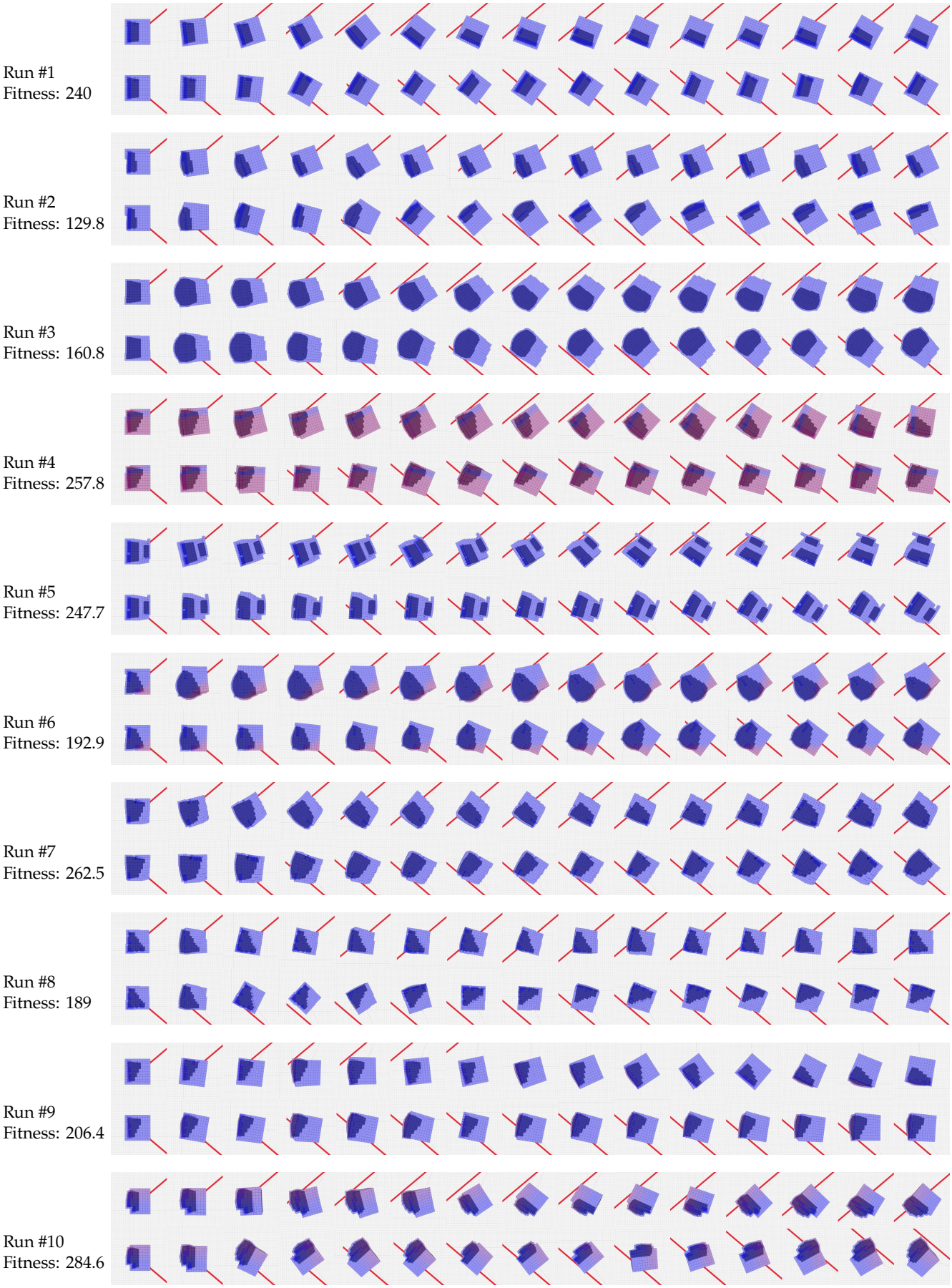


Figure A.6: Timelapse of the *dir-closed* champions seen from above (corresponds to fig. A.4, steering 40 degrees left and right in each pair of images. The red line represents the target direction. The robots are rendered transparently; the dark areas inside the robots are the pneumatic chambers. Material colour ranges from blue = hardest to red = softest. Each sequence spans 10 seconds. The fitness is calculated according to section 3.4.2. The run numbers are just labels and have no further meaning.

parameter	value	notes
pop_size	50	population size
init_recurrent_weights	random	how to initialize the weights for the recurrent nodes
activation_default	random	initialization of node activation function type
activation_mutate_rate	0.01	mutation rate of node activation function type
activation_options	sigmoid sin abs square exp clamped gauss	possible node activation functions
compatibility_disjoint_coefficient	2.0	disjoint and excess gene counts' contribution to genomic distance
compatibility_weight_coefficient	1.0	weight and bias contribution to genomic distance
compatibility_threshold	3.5	genomic distance threshold for species boundaries
conn_add_prob	0.0713	probability of adding a connection
conn_delete_prob	0.03	probability of removing a connection
node_add_prob	0.05	probability of adding a new node
node_delete_prob	0.0	probability of deleting a node
enabled_default	True	all connections start as enabled
enabled_mutate_rate	0.01	probability to mutate "enabled" status
num_inputs	4	number of input nodes
num_hidden	0	number of hidden nodes at genome initialization
num_outputs	4	number of output nodes
initial_connection	full	connection at genome initialization
weight_init_mean	0.0	weight initialization mean
weight_init_stdev	1.0	weight initialization standard deviation
weight_max_value	3.0	maximum weight value
weight_min_value	-3.0	minimum weight value
weight_mutate_power	0.0025	mutate power for weights
weight_mutate_rate	0.5	mutate rate for weights
bias_init_mean	0.0	bias initialization mean
bias_init_stdev	1.0	bias initialization standard deviation
bias_max_value	3.0	maximum bias value
bias_min_value	-3.0	minimum bias value
bias_mutate_power	0.0025	mutate power for biases
bias_mutate_rate	0.5	mutate rate for biases
max_stagnation	15	generations after which a species is considered stagnant
species_elitism	1	number of elites per species
tournament_size	2	size of tournament for selection
min_species_size	2	minimum number of individuals per species
weight_init_sd	3.0	standard deviation to initialize controller weights
weight_mut_init	3.0	initial value of the controller mutators
ctrl_hidden_nodes	4	number of hidden nodes in the controller

Table A.1: Hyperparameters of the NEAT algorithm