Design for an Increasingly Protean Machine

A Dissertation Presented

by

Sam Kriegman

to

The Faculty of the Graduate College

of

The University of Vermont

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

October, 2020

Defense Date: September 4th, 2020
Dissertation Examination Committee:

Josh Bongard, Ph.D., Advisor
Peter Spector, M.D., Chairperson
Nick Cheney, Ph.D.
Laurent Hébert-Dufresne, Ph.D.
Cynthia J. Forehand, Ph.D., Dean of Graduate College

Abstract

Data-driven, rather than hypothesis-driven, approaches to robot design are becoming increasingly widespread, but they remain narrowly focused on tuning the parameters of control software (neural network synaptic weights) inside an overwhelmingly static and presupposed body. Meanwhile, an efflorescence of new actuators and metamaterials continue to broaden the ways in which machines are free to move and morph, but they have yet to be adopted by useful robots because the design and control of metamorphosing body plans is extremely non-intuitive. This thesis unites these converging yet previously segregated technologies by automating the design of robots with physically malleable hardware, which we will refer to as protean machines, named after Proteus of Greek mythology.

This thesis begins by proposing an ontology of embodied agents, their physical features, and their potential ability to purposefully change each one in space and time. A series of experiments are then documented in which increasingly more of these features (structure, shape, and material properties) were allowed to vary across increasingly more timescales (evolution, development, and physiology), and collectively optimized to facilitate adaptive behavior in a simulated physical environment. The utility of increasingly protean machines is demonstrated by a concomitant increase in both the performance and robustness of the final, optimized system. This holds true even if its ability to change is temporarily removed by fabricating the system in reality, or by "canalization": the tendency for plasticity to be supplanted by good static traits (an inductive bias) for the current environment. Further, if physical flexibility is retained rather than canalized, it is shown how protean machines can, under certain conditions, achieve a form of hyper-robustness: the ability to self-edit their own anatomy to "undo" large deviations from the environments in which their control policy was originally optimized.

Some of the designs that evolved in simulation were manufactured in reality using hundreds of highly deformable silicone building blocks, yielding shapeshifting robots. Others were built entirely out of biological tissues, derived from pluripotent *Xenopus laevis* stem cells, yielding computer-designed organisms (dubbed "xenobots"). Overall, the results shed unique light on questions about the evolution of development, simulation-to-reality transfer of physical artifacts, and the capacity for bioengineering new organisms with useful functions.

Citations

Material from this dissertation has been published in the following form:

Kriegman, S. et al.. (2017). A minimal developmental model can increase evolvability in soft robots. In *Proceedings of the Genetic and Evolutionary Computation Conference*.

Kriegman, S. et al.. (2018). How morphological development can guide evolution. *Scientific Reports* **8** (1), 1–10.

Kriegman, S. et al.. (2018). Interoceptive robustness through environment-mediated morphological development. In *Proceedings of the Genetic and Evolutionary Computation Conference*.

Kriegman, S. et al.. (2019). Automated shapeshifting for function recovery in damaged robots. In *Proceedings of Robotics: Science and Systems*.

Kriegman, S.. (2020). Why virtual creatures matter. *Nature Machine Intelligence* **1** (10) 492.

Kriegman, S. et al.. (2020). Scalable sim-to-real transfer of soft robot designs. In *Proceedings of the IEEE International Conference on Soft Robotics*.

Kriegman, S. et al.. (2020). A scalable pipeline for designing reconfigurable organisms. *Proceedings of the National Academy of Sciences* **117** (4) 1853–1859.

Table of Contents

Chapter 1

Introduction

1.1 The Problem

> *The soles of our boots wear thin, but the soles of feet grow thick, the more we walk upon them.*
>
> —D'Arcy Thompson (1917)

This thesis has as its basis the fact that organisms are at once autonomous and adaptive systems, and robots are not.

By far the most commercially successful autonomous robot to date is the self-driving vacuum: the Roomba. Roombas are autonomous but non-adaptive systems. Mine is routinely defeated by the same long shoelace, which it eats and coils around its spinning brush and wheels until they grind to a halt. An adaptive robot would sense that something had gone awry, and respond by altering its behavior so as to preserve its functionality. That is, an adaptive robot would change so as to remain the same.

Autonomous adaptive machines could provide immense social utility. They could be used to patrol, remediate, and protect our bodies and our planet. They could help us build new infrastructure, here on Earth and in space, and technological artifacts far beyond the size scales we have achieved thus far. If successful, an increasingly dense, rich, and empowering web of human-robot interaction would be realized. Robots would no longer simply serve as workers or mere extensions of the human mind, like most other technologies. They would become our peers. However, this vision will only be possible if our robot symbionts are both autonomous and adaptive, like every organism on Earth.

One could argue that the essential difference between organisms and current robots is that organisms are *protean* systems: Organisms, like Proteus [184], are constantly forming and reforming. All conventional robots to date, on the other hand, are deployed with a fixed morphology which does not change during operation.

1.2 Contributions

> *Proteus will seek to foil you by taking the shape of every creature that moves on earth, and of water and of portentous fire; but you must hold him unflinchingly and you must press the harder.*
>
> —Homer (C8th B.C.)

In the pages that follow, I will demonstrate the utility of increasingly protean machines through a series of evolutionary robotics experiments that were performed over the course of four years, during my time as a Ph.D. student at the University of Vermont (2016-2020). Seven key benefits were uncovered and scrutinized:

1. Protean body plans permit continual learning [233] and open-ended evolution [148] in machines with open-loop or static control systems. This is possible because of hardware revisions, persisting over extended periods of time, that cause the same sequence of actions to generate different behaviors (chapters 2, 3, 6 and 7).

2. Protean body plans introduce arbitrarily small, and thus arbitrarily safe mutations [131], whose behavioral impact manifests temporarily during operation, rather than permanently from deployment (chapter 3). Evolution can then lengthen the time intervals containing superior traits and reduce the intervals of inferior traits. This allows evolution to surgically revise morphology with a pair of tweezers, if you will, rather than a sledgehammer.

3. Protean body plans smooth the search space evolution operates in via the Baldwin effect (chapters 3 and 4). This effect is outlined in section 1.8. In brief, the flexibility of a protean feature (e.g. skin thickness) allows evolution to sweep through design space along a line of development rather than sampling a single point of a morphologically-static agent. Development can then be incrementally canalized about a good static trait (e.g. calluses). This gradual homing-in process is hastened by natural selection, and eventually supplanted by the genetic determination of the feature (e.g. embryonic calluses). This well-studied phenomenon in artificial neural networks [96] is revisited with a twist: physically embodied robots, rather than an abstract control system. This distinction is important because it grounds hypotheses in the constraints and opportunities afforded by the physical world.

4. Protean body plans promote (the canalization of) permissive body plans: bodies that are robust to control changes (chapter 4). It is much easier to train a

control policy within a permissive body. This exposed a previously unknown detail about the Baldwin effect: instead of all useful traits becoming genetically assimilated, only traits that render the agent robust to changes in other traits become assimilated. I refer to this as *differential canalization*.

5. Protean body plans foster "zero-shot" generalization [220] (on the very first try, without adjustment) to previously-unseen static morphologies (chapter 5). The amount of generalization was found to depend on the kinds of interoceptive signals used to guide their morphological change during optimization.

6. Protean body plans amplify robustness to damage (chapter 6). This is because a sufficiently deformable body can in some cases morph to "resonate" with an existing control policy—tuned prior to damage—to regenerate behavioral competence. In some cases a shapeshifting robot can deform its remnant structure to regenerate (more or less) the original body shape. However that is not always possible or optimal given the remnant material and degrees of freedom. A shapeshifting robot can also form entirely novel shapes to recover function under a static control policy. This allows protean machines to recover functionality after "deep insult", such as removal of all limbs, or being cut in half.

7. Protean body plans can, in some cases, cohere the unpredictable behavior of biological building blocks (such as cardiomyocytes) into more predictable whole-body behavior (chapter 7). Evolution in silico can yield an appropriate static morphology that denoises and stabilizes even completely random control policies. Under certain constraints, such robust designs can be built from biological building blocks. This in effect can yield new organisms that are by virtue of their living cells inherently protean.

In addition to these seven intellectual contributions, two new platforms for constructing physical protean machines were developed as part of this thesis:

1. *Voxcraft.* A modular soft robot design and construction kit (used in chapters 2 and 6), voxcraft is open-source, safe, inexpensive, and simple. The design software is essentially a GPU-accelerated re-implementation of Voxcad [95], with a more salable tree-based collision system. The build protocol involves 1-axis rotational drip-molding, which was inspired by a 2-axis rotational molding technique [255]. The goal of voxcraft is threefold: Lower the barrier of entry to soft robotics for non-experts, facilitate new research in evolutionary robotics, and simplify the replication of those studies. More information can be found at: voxcraft.github.io

2. *Reconfigurable Organisms.* Chapter 7 reprises the 2020 study that introduced the first computer-designed organisms: xenobots. AI methods automatically design diverse candidate lifeforms in silico to perform some desired function, and transferable designs are then created using a cell-based construction toolkit to realize living systems with the predicted behaviors. More information can be found at: cdorgs.github.io

1.3 Ontology

> *A "system" is a set of variables sufficiently isolated to stay discassable while we discuss it.*
>
> —W. Ross Ashby

In the chapters that follow, we will incrementally liberate different physical attributes of a robot's design from the assumption that they are hand designed and fixed, so that they may vary during operation. This section defines each such variable, how they relate to each other (Fig. 1.1), as well as the different time scales in which they may change. The definitions are necessarily fuzzy around the edges, idealized, stipulative, and include (at least) one category mistake—but they will be precise enough to be useful. Later, when more exact concepts have been developed (e.g. how physics is simulated), it will be possible to state the ontology more precisely.

1. **Time.** It will be convenient to define $\{T \in \mathbb{N} \,|\, T > 0\}$ for evolutionary (phylogenetic) time, and $\{t \in \mathbb{R} \,|\, t > 0\}$ for developmental (ontogentic) time. Time is strictly positive, with t nested inside of T. Nested inside of t is another timescale—that of physiological functioning (e.g. the beating of a heart) and behavior (e.g. the cycle of a gait): "here and now" as Pfeifer and Bongard [187] put it. Time is of course relative to an observer—and there are nested observers. Here-and-now is minutes for a squirrel burying acorns, seconds for a dog chasing the squirrel, milliseconds for a racing heart, microseconds for its gated ion channels, pico- or femtoseconds for its vibrating molecular bonds. We can denote these finer timescales $\theta_1, \theta_2, \ldots$ but that will not be strictly necessary for the purposes of this thesis.

   T and t are sufficient to describe a basic synchronous evolutionary algorithm in time. For the entire n-th generation of evolution, $T = n$ as a population of designs (robot phenotypes) develop and behave for τ seconds during ontogeny: an evaluation period over $t \in (0, \tau)$. The worst designs are culled and replaced

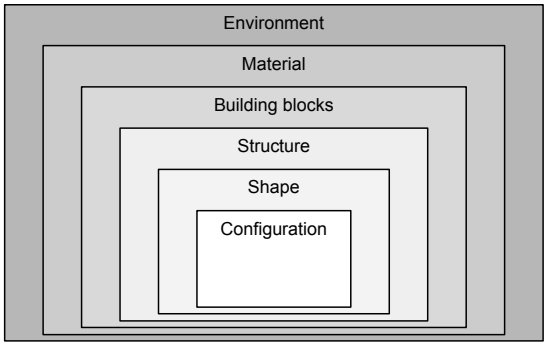


Figure 1.1: **Taxonomy of a robot's state.** The environment is a source of innumerable variables that can affect the robot. Within the environment there are materials from which building blocks may be fashioned. Building blocks of material are connected together to form a structure. A given structure can be deformed into various resting shapes. Configurations deflect the robot's structure about a resting shape, with elastic strain energy proportional to the deflection.

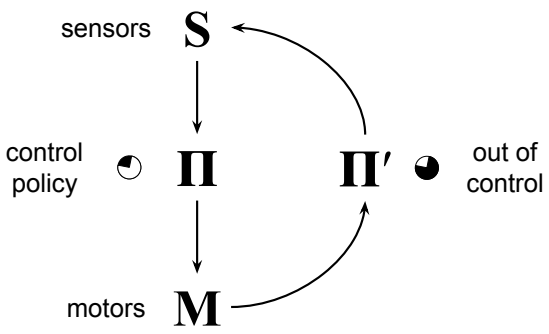


Figure 1.2: **Functional decomposition of state.** A set of sensors $\mathbf{S}$ are input to a policy $\mathbf{\Pi}$ which controls a set of motors $\mathbf{M}$. Motors influence the environment $\mathbf{\Pi}'$, sensors record the repercussions of actions. Traditionally, a robot's structure, shape and material properties are not varied by $\mathbf{\Pi}$: they are "out of control". They are, from the perspective of $\mathbf{\Pi}$, part of the environment. This Cartesian body/policy dissection is ubiquitous in AI, and sometimes useful for predicting behavior.

by noisy copies of the survivors. T is incremented by one, t starts over at 0, and generation $n + 1$ begins.

Note that this neat organization of timescales will later be turned inside out when designs are evolved without development in silico, and then built as organisms that develop but do not evolve and cannot reproduce in vivo.

2. **Material.** Embodied systems are made out of physical matter with mechanical properties—mass, density, friction, stickiness, viscosity, elasticity, plasticity, stability, strengths and limits—which can either streamline or frustrate the evolution of adaptive behavior [240]. Material properties can be heterogeneously distributed across a robot's body, and in some experiments, can change in both evolutionary and developmental time.

3. **Building block.** The smallest constituent unit of material that constitutes a robot. A block can be a passive slab of raw material, or it can itself be composed of functional subcomponents, circuitry, sensors and motors—but these sub-block-level elements cannot be manipulated by the designer or optimizer. In most of the following chapters, the building blocks will be elastic voxels, because the simulator employed for most of this work is voxel-based. Voxels are congruent cubes that attach face to face to form a structure, known formally as a polycube. In computer-designed organsims, however, the building blocks are biological cells: de/ciliated ectoderm and cardiomyocytes.

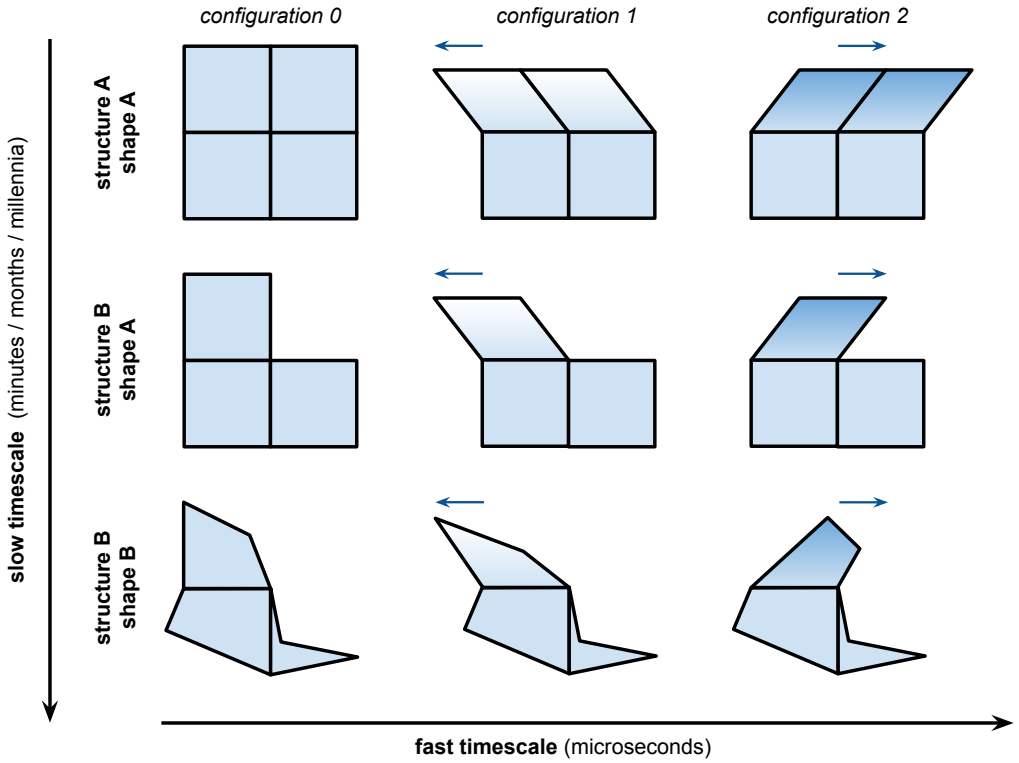


Figure 1.3: **Changes of geometry in time. By row, top to bottom:** A 2-by-2 grid of building blocks is generated (structure A). Each block has squarish resting shape (shape A) and identical material properties. The material details—and how they might change over time—are not depicted. A change in structure occurs when one of the blocks is subtracted from the body plan (structure B). Later, changes to the local volume and curvature of each block deform the overall geometry (shape B) without adding, removing or reorganizing the blocks (structure B). **By column, left to right:** The body at rest (configuration 0). When behaving, dorsal blocks are rapidly shifted back and forth, to the left (configuration 1) and to the right (configuration 2). Active blocks are shaded in proportion to their relative displacement due to configuration: lighter when angled to the left, and darker when angled to the right.

4. **Structure.** The contiguous topological arrangement and coupling of building blocks. The space of possible structures that can be built out of a pile of voxels—the set of polycubes—and the duplicates that result when these structures are translated, rotated or reflected—are enumerable. This simplifying constraint biases evolution toward regular, buildable structures and it makes it easier for any humans in the loop to think about the design space.

5. **Shape.** Plastic deformations and folding of the building blocks of a given structure, which persist after stresses have been removed. Or, elastic deformations that are held relatively constant under stress, within some interval of developmental time, during a sequence of configurations (definition 6) (e.g. as the leg

moves from point to point). Shape change can adjust the robot's posture, volume, mass distribution, number/placement of limbs, storage/release of elastic strain energy, surface contact geometry and tribology, during behavior.

6. **Configuration.** The displacement, orientation, and elastic deformation of a robot's resting shape during ontogeny. When the robot relaxes, it returns to its resting shape. For example, a spherical robot could deform to generate four limbs (shape change) and then move those limbs back and forth to gallop across flat terrain (configuration change). Later, the robot could extrude five clawed fingers at the end of each of its limbs. By moving its limbs and fingers and torso in a different pattern, it climbs a tree, and swings from branch to branch. As the robot travels through the different configurations of running or climbing or swinging, its body will stretch and contort about its root shape.

   For a protean machine that can intentionally change both shape and configuration, the distinction between the two becomes somewhat arbitrary. Are they not both simply "actuation"? This distinction is perhaps most secure in terms of the time scales on which processes of shape change and configuration change occur naturally. An amorphous blob could in principle continuously morph as a form of locomotion, rendering its "configuration" and "shape" equivalent. In nature, however, there is modularity: an underlying anatomical shape that is developing yet persists as the organism behaves (Fig 1.3).

7. **Environment.** Variables whose changes affect the robot, and variables which are changed by the robot's actions. This definition, adopted from Ashby [7], is intentionally broad, encompassing an infinitude of inanimate objects and forces, properties of the surrounding medium, the terrain, as well as other robots and living systems around and inside the robot.

8. **State.** A robot's environment, configuration, shape, structure and material properties, taken together, at any moment in ontogeny $0 < t < \tau$.

9. **Controller.** When thinking about robots, it is sometimes useful to define "controller" as a separate entity—a "policy" [227] that steers the robot along a "line of ontogeny": a succession of states. This is particularly helpful when dealing with electronics and software that is—for good reason—abstracted far away from the details of transistors and ions and sensorimotor circuits. But we must remind ourselves that this is, in fact, a categorical error [67, 90, 187].

   Without invoking a *Deus ex machina*, a line of ontogeny can only be formed by the robot's state and the condition of its immediate surroundings. Decomposing state into modules that on one hand "steer" and on the other "are steered", is

an example of what Bennett et al. [20] call a mereological[1] fallacy (e.g. "brains predict" [47]). Thus "controller" is formally identical (isomorphic) to "state" and redundant in our ontology. But it is nevertheless useful (Fig. 1.2).

10. **Field.** Borrowing yet another page from Ashby's book, a "field" is defined as "the phase-space containing all the lines of [ontogeny] found by releasing the system from all possible initial states in a particular set of surrounding conditions." We can now see "behaviors" as attractor states in the field of a robot arising from interactions among configuration, shape, structure, material properties, controller and environment.

    When different variables of a robot change on different timescales, slower moving variables define a subfield through which lines of faster moving variables may unfold. If a set of variables is fixed, the lines of ontogeny run in a subspace orthogonal to the axes (Fig. 1.4). Thus, the robot's current structure, material properties and resting shape define which configurations are available to the robot, and which are out of reach.

    Reconsider the example of limb growth in an initially spherical robot. The field surrounding an actuated sphere turns almost all lines of configuration change into rolling behavior. However, by stretching a few physical attributes past a critical point, the robot finds itself in a new field (Fig. 1.5), capable of previously unreachable behaviors (walking, climbing, brachiation).

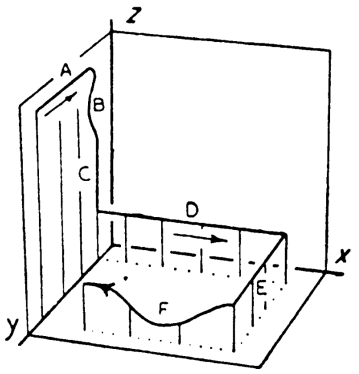


Figure 1.4: Ashby [7] illustrated how inactive variables restrict the active ones to an orthogonal subspace: "In the different stages the active variables are: A, y; B, y and z; C, z; D x; E y; F x and z." Variables can be practically inactive in relation to those that change on a much faster time scale.

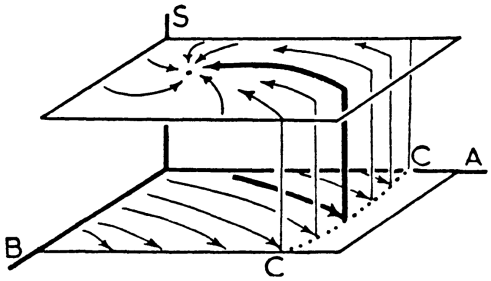


Figure 1.5: In describing variables due to step-mechanisms (S), Ashby [7] made clear how a system could be pushed onto new fields through critical states (from C to C). Morphological change (e.g. growing new effectors) can likewise drive a fixed sequence of actuation through disparate fields of configurations and novel behaviors.

[1]From the Greek $\mu\acute{\varepsilon}\rho o\varsigma$ *meros*: "a part, a fraction".

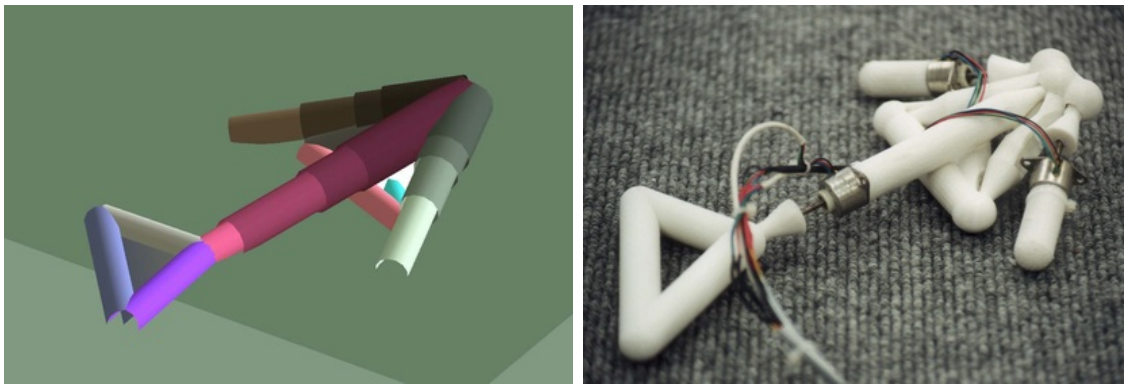


Figure 1.6: **Lipsonian AI.** An evolved simulated robot (left) and its 3D printed equivalent (right) [140].

1.4 Evolved Robots: The Fossil Record

In 1994, under the aegis of the Thinking Machines Corporation and the MIT Media Lab, the very first evolved yet virtual robots were reported by Sims [214, 215] and Ventrella [242]: structure and control of autonomous agents were co-optimized so that they ran, jumped, swam, performed phototaxis, and fought head-to-head for resources in a virtual world that followed (more or less) the laws of classical mechanics. The robots were relatively simple, composed of just a handful of jointed, rigid components, but their behavior was surprisingly rich and lifelike.

These experiments started with an objective, such as locomotion, and a population of randomly assembled robots. Although it was unlikely that any randomly assembled robot would fully satisfy the objective, by replacing the worst-performing designs with slightly- and randomly modified copies of the better ones, the population made incremental progress, generation by generation. It was the survival of the fittest, or, in the case of locomotion, the fastest.

Six years later, at Brandeis University, Lipson and Pollack [140] transferred similarly evolved robots from simulation to reality (Fig. 1.6) with a technology just then emerging: 3D printing. Robot designs were rapidly and safely prototyped as virtual creatures, discarding the truly awful or dangerous designs before testing them in reality. The result were "robotic lifeforms" that were designed, optimized, and built, end-to-end, with almost no human intervention.

Although Sims, Ventrella, Lipson and Pollack, and many, many others [9–11, 26, 27, 30, 33, 38, 40, 42–44, 46, 94, 98, 100, 114, 115, 130, 132, 133, 137, 182, 193, 197, 211, 240, 243] used an Evolutionary Algorithm (EA) to design simulated and/or physical robots, evolutionary robotics is independent of the specific procedure used to automatically generate and test designs.[2] EAs happen to be particularly

[2]Evolutionary robotics can be evolutionary robotics without an Evolutionary Algorithm because

useful in this domain because, apart from being exceedingly simple and parallelizable (Algorithm 1), few assumptions must be made about the robot *a priori*. The robot's material properties, building blocks, structure, shape, configuration, and controller can all be optimized together. For 25 years, structural optimization proper—changing the number and placement of mechanical degrees of freedom (building blocks), not merely tuning the parameters of a predefined structure—had only been achieved through evolutionary design.

1.5 Evolutionary Algorithms

Evolutionary Algorithms [48, 78, 79, 91, 172] differ from "gradient-based" approaches to optimization in two important respects. First, EAs require a population of agents (candidate designs) spread out on the fitness landscape,[3] whereas learning occurs within a single agent. Second, individuals in an evolving population are modified through random mutation, without any anticipated benefit. Gradient-based learning algorithms, in contrast, predict the repercussions of design revisions before applying them; they estimate the local topography (the gradients) of the fitness landscape around the current design, so that revisions can be applied that are expected to improve the design (climb the gradient). Evolution has no such foresight.

Algorithm 1: Evolution.

```
1: P ← CreateRandomPopulation()
2: while not TerminationCondition() do
3:     EvaluatePopulation(P)          ▷ n individuals evaluated at once on n threads
4:     S ← SelectParents(P)
5:     P ← RecombineAndMutate(S)      ▷ appends or replaces S with mutants
6: end while
7: return GetFittestIndividual(P)
```

any design process inevitability involves an evolutionary process of generate-and-test [61] on one spatiotemporal scale or another. Pathak et al. [180] used an algorithm inspired by Skinnerian development rather than Darwinian evolution to automate robot design—and did not refer to their work as evolutionary robotics. But as Fields and Levin [75] note, "the evolution of life on earth can also be seen as the development of life on earth... [as] all descendants of [the last universal common ancestor] together compose one continuous cell lineage, that is, one biological individual."

[3] A "fitness landscape" is a metaphorical terrain used to conceptualize multidimensional design space in terms of more familiar notions of 2D or 3D topographical space. Design variants are mapped to coordinates on the horizontal axis (Fig. 1.8) or plane (Fig. 1.9). The elevation of this landscape is fitness (higher is better). Unlike a phase-space or field, which describes the tendency of state transitions in ontogeny, a fitness landscape describes changes in fitness due to changes in design. There is no guarantee that mutations exist to move a design up a slope of fitness.

1.6 The March of Progress

In 2019, Pathak et al. [180] demonstrated how a robot's ontogenetic structure and configuration can be co-optimized using a gradient-based reinforcement learning (RL) algorithm, without evolution. Instead of evolving a monolithic morphology, Pathak et al. optimized a swarm of elemental agents—autonomous "limbs"—that, in addition to actuating, could choose at every timestep to either attach to their nearest neighbor (forming an aggregate, symbiotic machine with a shared reward function) or detach, reconfigure, and test a new design.[4] Thus structure was controlled by the same policy that coordinated configuration changes.

Despite not using a certified Evolutionary Algorithm, this too is evolutionary robotics. Indeed, the line between RL and EAs is becoming so thin, the two are almost indistinguishable. There are EAs that act more like a single agent learning than a population evolving [201]. And with the rise of massively parallel computer architectures, increasing interest in RL is coming to bear on population-based methods that can fill up those parallel threads with multiple learners [106]. What's clear is that Pathak's virtual robots evolved in the general sense (if not the stricter Darwinian interpretation) of the word "evolution". If we compared the structures assembled by randomly generated limbs to those assembled by fully trained limbs, and sampled some results along the way, we would get something like Rudolph Zallinger's iconic "March of Progress" from Dryopithecus to Modern Man, that inevitable stock image of evolution. There is no march of progress for non-evolved robots because the design is held fixed during training.

However, there has not been much of a march of progress for evolved robots either. Despite vast increases in computational power since 1994 and major advances in additive manufacturing after 2000, evolutionary robotics has floundered in a sea of Sims-like virtual creatures (Fig. 1.7) and Lipsonian machines (Fig. 1.6). In the two decades after Lipson and Pollack [140], very few [33, 38, 70, 94, 99, 182] have endeavored to evolve physical robots. Even in silico, free from most real world constraints, scant progress has been made scaling the complexity and competence of virtual robots [45] (Fig. 1.7).

1.7 The Evolvability of Robots

One reason that evolving robots has proven so challenging is the very fact that robots are embodied and situated. Local structural revisions, such as adding a fifth rotor to a functioning quadcopter, tend to have global behavioral repercussions. It is usually difficult to find a series of small adjustments that result in increasingly better design.

[4] Simply put, attaching/detaching was in the "action space" of each limb [227].

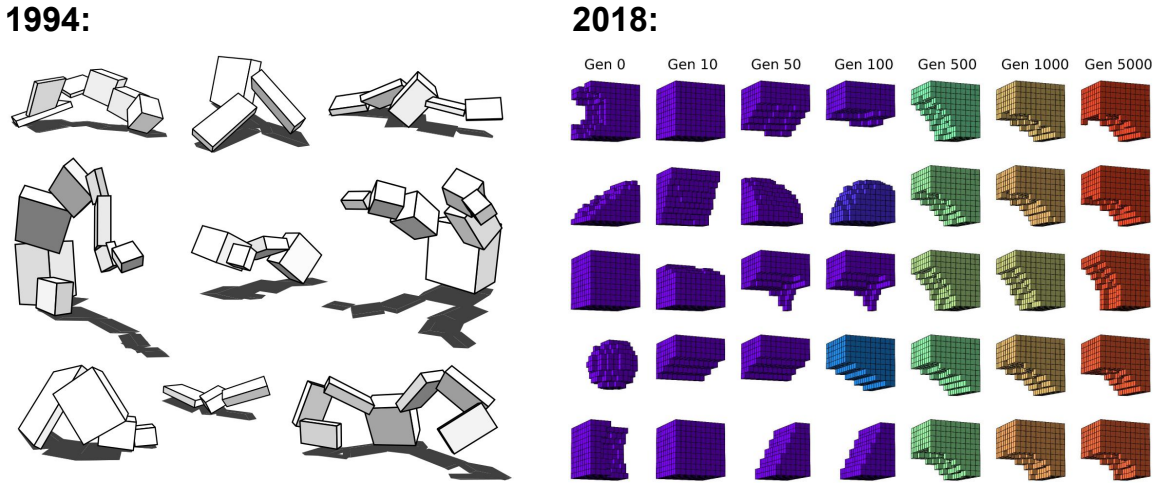


Figure 1.7: The first evolved simulated robots (Sims, [215]), and the state of the art (Cheney et al., [46]).

This property makes physical artifacts much more difficult to optimize than disembodied systems such as deep neural networks, which tend to preserve their behavior in the face of sweeping global revisions.[5]

Certain aspects of robot behavior are generated by the precise orchestration of its parts. When more parts are added from design revisions or subtracted from wear and tear, the mapping from sensors to motors—the control policy (Π in Fig. 1.2)—could be rendered more or less obsolete for the current task environment [27, 55].

Other aspects of behavior are generated by virtue of embodiment and the laws of physics alone, such as rolling down a declined plane, and the swing phase of human walking [50]. From the controller's perspective, such behaviors, or aspects thereof, are passive: the body and environment provide them "for free" (Π' in Fig. 1.2). The control system need not spend any of its valuable resources orchestrating this behavior from scratch; instead, it may exploit these passive dynamics, gently guiding them when necessary, to render useful work [240]. However, if control is predicated on particular body-environment interactions occurring automatically, and the body or environment change and disrupt this interplay, the controller will need to carefully redistribute its computational resources to steer these additional interactions while maintaining other aspects of behavior with less computational oversight.

[5]The most common method for training artificial neural nets is backpropagation of error, which estimates the gradient with respect to each weight of the network. Iterating backward, one layer at a time, error is propagated by chain-rule to upstream weights, which are adjusted in proportion to their respective gradient. Depending on the size of the network, thousands or millions (or hundreds of billions [191]) of parameters are all perturbed at once.

The behavioral consequences of physical mutations or damage are particularly harsh when morphology is ontogenetically-static, because any change to behavior manifests instantly and permanently from deployment or time of damage. It is usually easier to achieve acceptable levels of fitness by holding the body plan fixed throughout optimization and focusing solely on learning better controllers for the given body, regardless of the body's fitness potential [45]. Thus, much work has attempted to reduce the behavioral repercussions of morphological mutations [46]. A natural way to reduce the behavioral impact of a mutation is to spread that impact across a period of time: development.

1.8 The Evolution of Development

In a population of evolving but non-developmental robots, mutations can only change the physical layout by discrete steps, from parent to child. This is because the design of a non-developmental robot cannot change during its lifetime. The morphological and thus behavioral impacts of such mutations take hold immediately, at $t = 0$. Introducing development allows mutations to manifest slowly during the lifetime of a robot, or at the very end of the lifetime, which enables arbitrarily small design modifications.

Of course there are other ways to reduce the behavioral impact of mutations. For one, we could simply restrict the magnitude of morphological mutations to be vanishingly small. This might very well be sufficient given enough time, and if the fitness landscape is smooth enough to ensure that a sequence of tiny steps will progressively increase fitness. However, the search space of possible robot designs tends not to be smooth in realistic settings; rather than hills and gentle slopes, the landscape is typically full of cliffs and harsh spikes.

Some great designs, we may suppose, are only of value when perfect or nearly so; changing them by even a single bit is all but certain to entirely break their functionality. In other words, there is no gradient—no sequence of mutations—leading toward these great designs. In such a space, a near miss is as good as a mile, and evolution can be no better than random search. However, as Hinton and Nowlan [96] so neatly demonstrated: development can, under certain conditions, smooth the search space that evolution operates in, thereby creating a gradient toward increasingly better mutations (Fig. 1.8).

This process, known as the Baldwin effect [12, 62, 213], is possible because development sweeps over several traits in a single agent, and sometimes exposes promising static traits along the way. This can create a new gradient in the evolutionary search space, rewarding descendants that more rapidly manifest the good trait during their lifetimes (and retain it through the remainder of their lifetime). Following the gra-

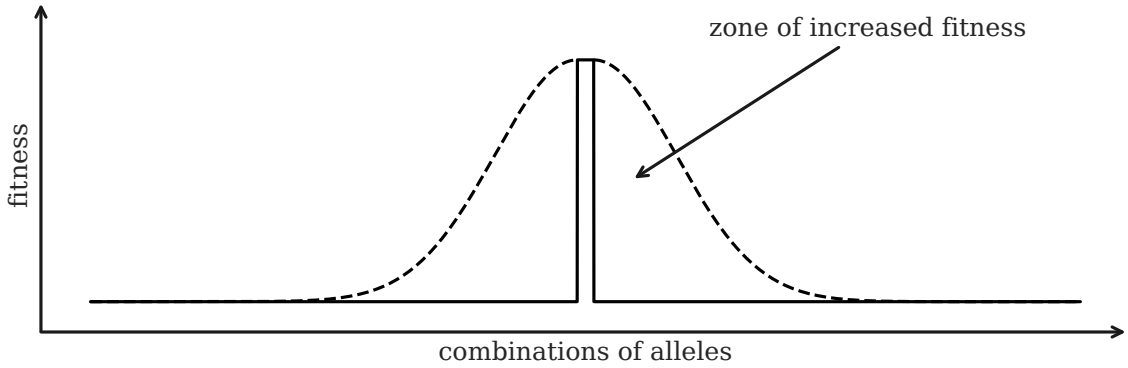


Figure 1.8: **The Baldwin Effect.** Hinton and Nowlan [96] considered the evolution of a bitstring that is only of value when perfectly matching a predefined target string. The search space of possible combinations of alleles (bit values) therefore has a single spike of high fitness with no slope leading to the summit (solid black line). "The good [string] is like a needle in a haystack." Here's where development, through the so-called "Baldwin effect", can potential guide evolution: Suppose that the "lifetime" of each bitstring is lengthened from a single point to a period of time in which a portion of the string can change. Then there is a chance that some string in flux, eventually, flips the right bits and stumbles upon the good static string during its life. If evolution controls which portions of the string are flippable and which are non-flippable (i.e. genetically determined), and the later are all correctly specified, then the speed at which such individuals tend to hit the target will be proportional to the number of non-flippable bits. Finally, it is reasonable to assume that the faster an individual discovers the good design during its lifetime, the more fitness it accrues. Taken together, this has the effect of creating a gradient of increasing fitness (dotted black line), leading up to the correct specification, that natural selection can climb by incrementally hardcoding more correct bits in the genotype. "It is like searching for a needle in a haystack when someone tells you when you are getting close." (Redrawn from [96].)

dient requires a series of mutations that incrementally reduce development such that the good trait becomes "locked-in" in more morphologically-static descendants [244]. Development is thus a ladder that, like Wittgenstein's, can be thrown away once it has been climbed. However this is only possible if such mutations exist.

When a behavioral change manifests late in the lifetime of a developing robot, it might discover a promising static design at the end of its life. The sequence of mutations required to prepone or "earlify" this trait such that it arises increasingly earlier in the lifetimes of descendants, might be hard to find or might not exist at all. Symmetrically, if a promising static design is discovered early in the lifetime of a robot, but the robot develops out of it, the hypothetical series of mutations that prolong this behavior may be illusory or impossible to execute without completely disrupting behavior. Indeed, one of the first experiments to explore the interaction between evolution and (neurological) development in controllers onboard real robots [77], found that such mutations are unlikely to exist if agents exploit developmental change for behavior.

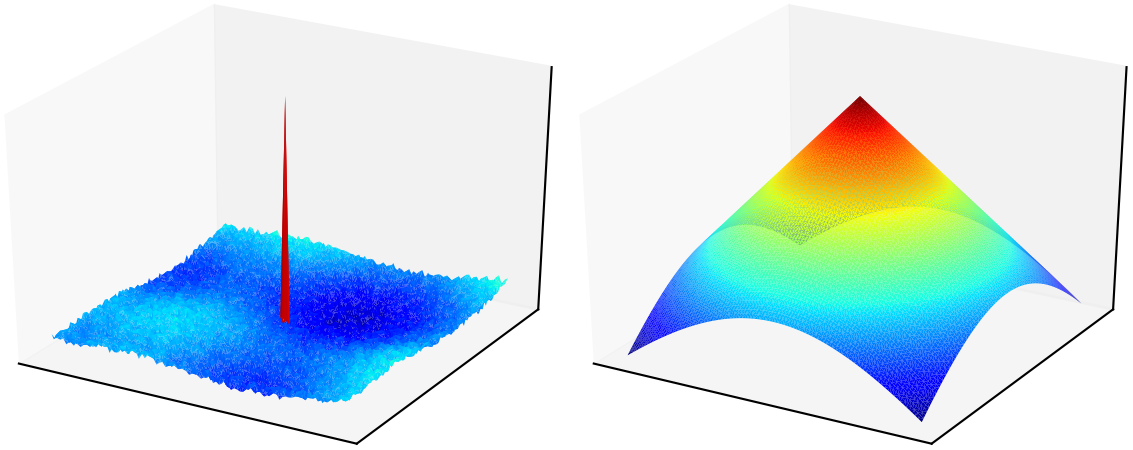


Figure 1.9: **Two sides of the same space.** In each plot, points on the horizontal axes directly map onto a 2D continuous design space. For example, x could encode the initial elastic modulus of a bipedal robot's two limbs, and y might then indicate their initial placement along the circumference of its spherical trunk. The vertical axis is fitness. On the left, we have a highly-fit genotype (red spike) among a sea of designs with much lower fitness (blue waves). The red design might be an upright bipedal walker with the appropriate elasticity, while the blue designs perhaps can only scoot, tumble, or row themselves forward (if they can move at all). Evolution cannot see the red spike, as we can. Evolution only "sees" the relative elevation beneath each design in the current population. On the right: the aftermath of the Baldwin effect. Developmental plasticity alters the space by introducing a gradient: Proximity to the great design determines the speed and reliability with which it may be expressed during development, and thus determines (expected) fitness.

When picturing Hinton and Nolwan's idealization as a single column of high fitness in the middle of an abstract 1D design space (Fig. 1.8), it can be difficult to imagine how evolution moves through a real, multidimensional search space. So at the risk of being redundant, I have included a second depiction of the Baldwin effect (Fig. 1.9) in which the vertical axis is again fitness, but in which the horizontal axes directly map onto continuous parameters of the design. Development unfolds on a line in this space, with the vast majority of possible trajectories missing the good trait combination (the red spike of high fitness in Fig. 1.9).[6] Evolution discretely samples the space by setting the initial conditions (starting point) and how development may unfold across the landscape (curvature and length of the line) in time (acceleration along the curve) until death. Once found, successful developmental trajectories can become canalized by evolution such that the good trait combination becomes increasingly likely in future descendants. In the limit, the line of development shrinks to a point:

[6]There are typically many such (potentially Baldwinifiable) local optima, and so there is also evolution: a population spread out on the landscape, not just a single developmental robot.

the initial conditions.

1.9 Evolved Development in Robots

Hinton and Nowlan [96], and the cascade of subsequent computational/analytical studies to investigate this interplay of evolution and development, considered either abstract phenotypes [2, 5, 6, 19, 66, 74, 80, 150, 173, 202, 206, 228, 238] or control systems onboard morphologically-static hardware [77, 102]. Evolution and development could not modify body plans because, without exception, the body plan (if it existed) was fixed.

Several computational but embodied models of prenatal development have been forwarded [14, 29, 59, 60, 64, 65, 68, 109, 154, 160]. However, because development occurred prior to the evaluation period, as a kind of artificial embryology, the behavioral impact of a mutation still kicked-in at "birth" ($t = 0$), and so the Baldwin effect could not occur.

Prior to the work documented in this thesis, there were only three cases reported in the literature in which a simulated robot's body was allowed to change while it was behaving. In the first two cases [114, 243], it was not clear whether this ontogenetic morphological change influenced the evolution of behavior in any way. Later, Bongard [28] demonstrated how such change could lead to a form of self-scaffolding that smoothed the fitness landscape, not unlike the Baldwin effect. But the Baldwin effect did not occur, because it could not occur.

Bongard's experiments began on familiar ground. Populations of neural controllers were evolved to steer (the configuration of) a simulated quadruped/hexapod so that it moved toward a simulated light source. But this robot had a developmental trick inside each of its four (or six) limbs. The robot was, for most of its evolutionary history, deployed with an anguilliform body plan—an eel with an actuated spine. As the initially legless robot behaved (varied its configuration), limbs were slowly extruded from its trunk. As the limbs lengthened, they dipped their resting angle about the shoulder joint, transitioning the robot from a prone to upright stance, during development (shape change).

Early in evolution, limb growth occurred over the robot's entire evaluation period. In a series of four evolutionary stages, the rate of extrusion was incrementally increased such that the final, upright legged body plan was realized increasingly earlier in developmental time, and held fixed for the remainder of the robot's lifetime. In the final stage of evolution, the robot was deployed with its final shape from the very beginning of its behavior; development had been completely canalized.

Thus, by the end of evolution, these robots were identical to a control treatment but for a single isolated difference: one group had ancestors that developed, and the

other had ancestors which did not develop. However, this clean experimental design came at a cost: the robot's initial shape, and the way in which its shape developed, were both manually fixed *a priori*. Thus evolution could not modify morphological development, indeed, it could not modify morphology at all.[7]

1.10 Resilient Machines

> *Only variety can destroy variety.*
>
> —W. Ross Ashby (1956)

It is often the case that an artificial system may be trained to achieve high fitness in one environment, but fail spectacularly under even the slightest perturbations [8, 37]. By temporarily permitting shape to vary during training, the final, manually-canalized robot from Bongard's experiment [28] exhibited increased robustness in novel environmental conditions (random external perturbations). Shape change thus acted as a form of regularization: the robot had to maintain phototaxis while changing its body. This increased breadth of sensor-motor contingencies prevented the robot from overfitting its controller (of configuration) to the idiosyncrasies of a small sample of circumstances [108]. The upshot was robustness by virtue of a good static structure.

However, for a system to remain viable as its environment varies, the system must also be able to vary itself. Only variety can destroy variety, as Ashby put it. Thermostats, centrifugal governors, and Ashby's Homeostat[8] are examples of such systems—regulators—with the "requisite variety" to absorb and eliminate variety in the system being regulated. The field of each of these regulators is "ultrastable", as

[7] The linear-actuators-as-semipermanent-body-segments trick was also employed by a later robot to automatically adjust leg length in situ for different supply voltages [174] and terrain [175]. However, leg length did not vary during behavior (locomotion), so the Baldwin effect could not occur.

[8] The homeostat was built by Ashby in the late 1940s using surplus military equipment. It consisted of four identical electrically-coupled units. Each unit sent its output to, and received an input from each of the other three. A pivoted magnet sat atop each unit. DC output was proportional to the angular deviation of the unit's magnet from its central position. The magnet's deviation was in turn a function of input, internal circuitry, and adjustable parameters supplied by stepping switches. When current exceeded a certain value, the switches were energized, moving the parameters to new values. The field (defined in section 1.3) of the Homeostat's four variables (magnet positions) had only one state of equilibrium (centered), which was either stable or unstable. When the magnet's position was perturbed externally by mechanical force, the system would cycle through switch settings according to a random number table until it found a stable field and reached equilibrium, where it would actively resist subsequent external perturbations.

Ashby would say, tending all the time towards stabilization and physiologic constancy. However, these regulators are controllers (Fig. 1.2), not robots.[9]

In 2006, Bongard et al. [27] reported a "resilient machine" that, through continuous self-modeling, could vary its behavior to compensate for structural loss due to damage. The details are not important for the story I have chosen to tell about increasingly protean machines, beyond noting the relationship between regulators and models, which was made clear by Conant and Ashby [51]:

> . . . success in regulation implies that a sufficiently similar model must have been built, whether it was done explicitly, or simply developed as the regulator was improved.

In this case, explicit body schema were built from the ground up, in silico, through sensorimotor experiment in the physical robot. Because the robot modeled its structure, rather than just its behavior, it could absorb variation in both. Later, Cully et al. [55] demonstrated how a robot could in some cases do this much faster. However in both approaches, the mechanical details of the robot (structure/material/shape) were presented to the control system as a *fait accompli*, limiting the depth of insult from which such machines can recover.

These two studies were intended as proofs of concept—basic science that will undoubtedly be refined with additional research and development. In the meantime, they seem to have spelled out the limitations of clever software trapped inside non-protean machines. Such behaviorally-plastic yet morphologically-static robots can recover masterfully from certain shallow insults, such as the loss of a single leg to injury. But they do not have the requisite variety to recover from deeper mechanical damage, such as the removal of all limbs or being cut into a hundred smaller pieces.

The resiliency of such robots is limited not only because they cannot re-generate structure, motors or sensors—but because they cannot generate these body parts in the first place. Their ability to perceive the environment, and act against it, is constrained within a fixed set of perceptual categories and action alternatives, rather than open-ended. This shackles the robot's epistemic autonomy [36]: the robot's overall capacity for knowledge and cognitive success beyond what the designer predicted to be relevant for the problem at hand.

[9] Any system that cannot move through the environment has at best a questionable claim to be a robot.

1.11 A Protean Machine?

> *And so it was that two very tired young men trailed a microphone down into Baker Street from the upstairs window, and picked up the random noise of dawn traffic in the street. I was leaning out of the window, while Gordon studied the cell. "It's growing an ear", he said solemnly* (ipsissima verba).
>
> —Stafford Beer (2001)

There is a Cartesian bias in modern AI of presupposing a body plan and then building (ever deeper, hierarchical and non-linear) mappings between its fixed percepts (sensors) and fixed action space (motors) [129]. Increasingly protean machines can, as we will see, increasingly determine their own categories of perception and action, and thus position themselves at a continually greater distance from the bias, intuition and cognitive limits of their human designer. What follows is the story of the first, and some would argue the only truly protean artificial machine.

Paskian AI

In 1956, on Baker Street in the heart of London, Gordon Pask and Stafford Beer grew an electrochemical ear.[10]

Platinum wire electrodes were immersed in a shallow acrylic dish containing an acidic aqueous metal-salt solution [177] (Fig. 1.10). The flow of electricity through the solution caused the deposition of metal on the floor of the dish along lines of maximum current (Fig. 1.11). Metal deposits in effect extended the electrodes, generating circuits *ab initio*, instead of using components with their own well-defined forms and functions. In this electrochemical soup, there was only raw material: metal ions.

As metallic threads were being built out of ions on relatively negative electrodes, the acid worked to dissolve the threads back into ions. Threads that kept pace with the acid back reaction, drew more current down them due to their very low resistance (relative to the solution) and were thus reinforced and extended with additional metal. Growth was not linear, however. Each thread was fringed by short thin tendrils that extended into the solution in all directions, testing the waters as it were, before retracting back toward the main branch in perpetual dis/integration. Under a constant input set of voltages, threads competed for current, grew and dissolved, broke and

[10]The basic electrochemical system had been developed earlier in collaboration with A. Addison, a member of Heinz von Foerster's laboratory at the University of Illinois ([179], p. 107).

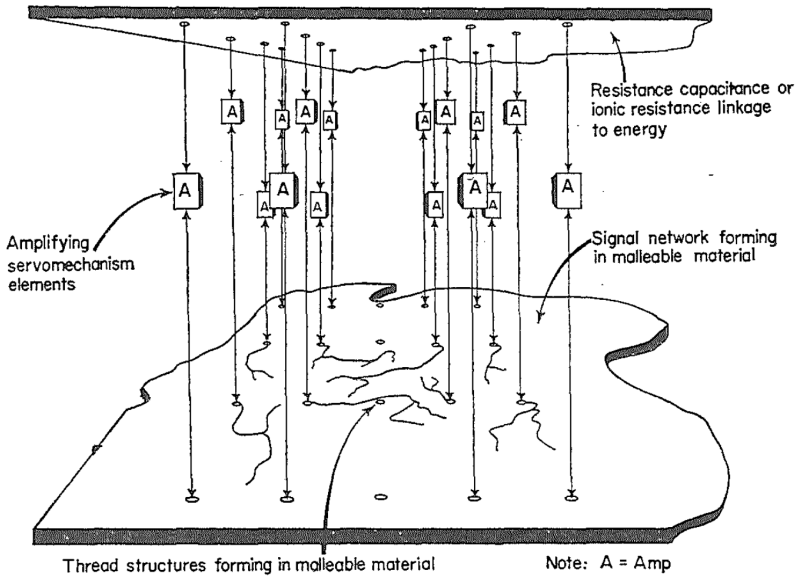


Figure 1.10: Electricity flows through an aqueous solution of iron salts, depositing low resistance metallic threads and influencing the subsequent flow of the current and metal (Pask [178]).

regenerated, bifurcated and sometimes combined before settling into a dynamically stable network with asymmetrical structure and robustness.

The electrochemical assemblage also displayed a kind of memory. When the input set of voltages were held constant, a big dense metallic tree grew steadily until it reached a stable structure. If the input set was then changed, the tree slowly shifted and restructured itself. If the input was then set back to the original distribution, the tree regenerated the initial structure, but this time stability was achieved much more quickly. This was because dissolution was gradual and residual particles helped subsequent threads grow back rapidly along the same lines. The longer a tree had been stably growing, the slower it disintegrated when the input set of voltages changed, and the quicker it returned to its original structure when the inputs were reset. That is, the robustness of the tree was determined by the conditioning time: reinforcement learning was possible.

A sensor electrode was dipped into the solution. The generated waveform of the electrical output could be conditioned by rewarding thread networks with an increase in current supply, which released an influx of free metal ions—the building blocks of threads. As a result, certain types of structures were allowed to survive and grow while those that did not act appropriately could be starved of ions and tended to die off. Pask noted that, regardless of how the electrodes were configured, the system would tend to develop a thread structure that led to current flowing in such a way that it was rewarded further. It is important to note that this reward is simply

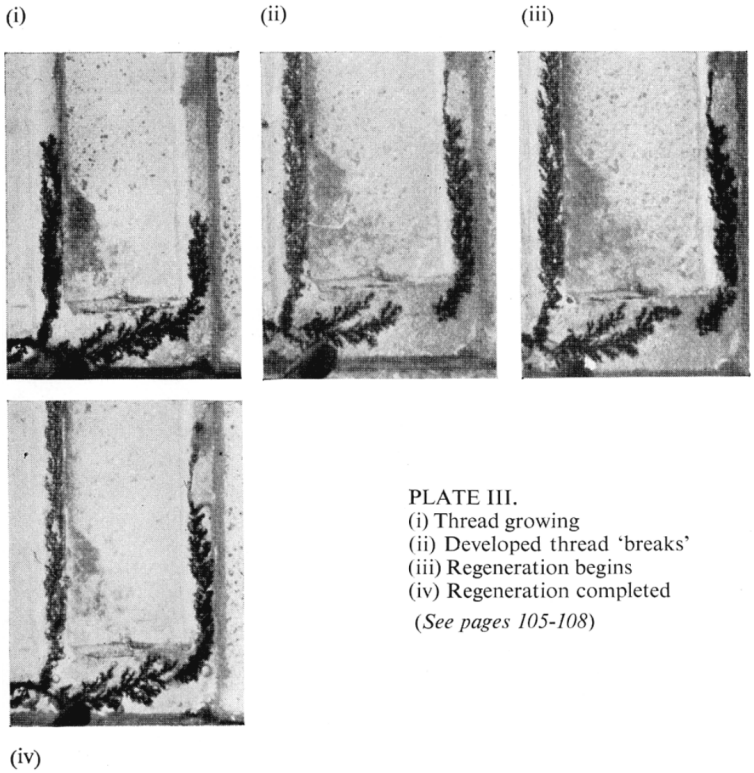


Figure 1.11: Dendritic threads of metal grow toward an electrode in iron-sulfate solution (Pask [179]).

an increased capacity for growth—there is no specification of what form the growth should take.

Thread networks could be steered and selected to become sensitive to a wide variety of stimuli impinging on their structure. Electrical output was noticeably affected by all sorts of variables, including, but not limited to temperature, pH, magnetic fields, and vibrations. So Beer and Pask took a microphone, hung it out of a window on Baker Street and picked up noise from the street. The electrochemical soup started to vibrate, and with reward, started to form a function in response to audio signals.

The device was later grown to recognize specific frequencies, as from a buzzer. Whenever a buzzer was sounded, if the electrical frequency of the buzzer was partially replicated at the sensor electrode, then the system was rewarded with its metal ions. By rewarding structures whose output criteria correlated with specific input criteria (the buzzer sound), the system became better at recognising the buzzer. Importantly though, by changing the input criteria, say by using electromagnetic fields rather than vibration, the system could dynamically grow a new type of sensor.

In proceedings of an Interdisciplinary Conference on Self-Organizing Systems

(May, 1959), Pask's paper [178] includes a transcribed discussion from his presentation. Responding to Alan Newell and John McCarthy, Pask reported two kinds of devices that were grown out of metal ions:

> We have made an ear and we have made a magnetic receptor. The ear can discriminate two frequencies, one of the order of fifty cycles per second and the other of the order of one hundred cycles per second. The "training" procedure takes approximately half a day and once having got the ability to recognize sound at all, the ability to recognize and discriminate two sounds comes more rapidly... The ear, incidentally, looks rather like an ear. It is a gap in the thread structure in which you have fibrils which resonate with the excitation frequency.

Pask's assemblage built up initially functionless elements (metal ions) into structures that could examine initially unspecified attributes of their surroundings: vibrations or magnetic fields. The machine had epistemic autonomy relative to its designer (Pask) because it was not bounded by a closed set of perceptual categories ("filters"). Instead, the machine was free to find its own "relevance criteria" in the world, and construct new kinds of sensors that were increasingly sensitive to it [36]. According to Pask [179]:

> By definition, intent and design this cannot occur in an artifact made from well-specified components.

Only a somewhat ill-specified system—a more or less protean machine—can achieve epistemic autonomy. This is important because evolution is essentially a hill climbing process, and as Pask noted:

> When the active elements of a hill climber meet an insoluble problem, the uncertainty about which of several possible actions to take is resolved by a dice throw. The thread, faced with the same dilemma, must become one kind of thing or another – there is no finite set of *possibilities* to choose between – and from the observer's viewpoint a structural uncertainty is resolved. This is precisely the behaviour remarked upon by the earlier embryologists – that development of a cell along a quantitative gradient gave rise to qualitative change.

When Beer [18] looked back on his "filigree friendship" with Pask, four decades after their electrochemical experiments on Baker Street, he did not mince his words explaining the significance of the first protean machine:

> This was the first demonstration either of us had seen of an artificial system's potential to recognize a filter which would be conducive to its own survival and

> to incorporate that filter into its own organization. . . [T]his facility would transform the world of information technology, if it could ever forget and transcend its origins in mere data processing. But that would require the overthrow of yet another paradigm.

1.12 Increasingly Protean Machines

By virtue of greater and greater morphological autonomy, each body plan in a sequence of increasingly protean machines possesses a greater capacity for robustness, agency and epistemic autonomy than any of the less protean body plans preceding it. However, optimizing a system whose sensor/motor modalities are continuously forming and reforming poses several ill-defined intellectual and technical problems. If the current paradigm of Cartesian AI is to be overthrown in favor of a more Paskian approach, additional empirical studies will need to be undertaken to more precisely articulate these problems and their potential solutions. This section traces the literature from Pask [177] to modern protean machines through the lens of the present thesis.

After Pask [177], two artificial systems were reported in the literature that evolved de novo sensitivity using hardware that was reconfigurable in situ. Both were conceived and built at the University of Sussex. In 1995, Thompson [232] evolved a discriminator of 1 kHz and 10 kHz square wave inputs on a small corner of a field-programmable gate array (FPGA), without a timer.[11] Upon dissection, the discriminator seemed to exploit strange recurrent loops, and parasitic capacitance between unconnected components (blocks of transistors). But no one truly knows how it worked. A few years later, Bird and Layzell [22] evolved a radio—by accident. They set out to evolve an oscillator using a network of transistors, but the solution ended up finding and utilizing radio waves emanating from nearby PCs. But, again, no one fully understands how the system was interacting with its environment. For instance, one of the evolved oscillators worked successfully until a soldering iron at a different workstation in the lab was disconnected from a wall outlet.

Our evolutionary history, lifetime experiences, and the way we perceive space and time makes it difficult for us to understand and optimize systems that are not well explained by human-scale mechanics, metaphors, or mathematical formalisms that impose somewhat stifling assumptions. The utility of protean circuits is therefore

[11] In conventional circuits, timing is supplied by capacitors, which require a constant amount of time to charge, through a resistor, until discharge. Charge will also flow back and forth between the plates of a capacitor, through an inductor, forming an electrical analogue of a tuning fork: a tuning circuit. Design problems that involve timing are nontrivial in a network of transistors—or using a FPGA, which implements combinational logic using groups of transistors—without resistor-capacitor networks, tuning circuits, oscillators, resonators, timers, counters, clocks or crystals.

clear: the evolution of such systems introduces diverse creative designs beyond the realm of human intuition. However, one could argue that these more modern systems were less protean than Pask's free-flowing metal structures, because their autonomy was restricted to flipping a fixed set of switches on a fixed circuit board.

Other modern technologies exist with much more morphological flexibility than Thompson's FPGA and Bird's radio. For instance, modular robots have been reported that self-assembled, -disintegrated, and -replicated by changing the arrangement of a fixed set of blocks [136, 196, 251, 257]; origami robots that self-folded [73, 83, 92, 118, 156]; soft robots that underwent plastic deformation or semipermanent elastic deformation [207–210]; and jamming robots that varied their material stiffness [34, 167, 224]. Among these platforms are marvelous feats of creativity and engineering. However, they lacked ontogenetic morphological autonomy relative to a human operator. They were not *free* to vary their body plan. They were forced to change according to an externally prescribed developmental schedule.

With our emphasis on autonomy, Pask's electrochemical ear arguably remains the most protean machine built to date. But how protean was it, really? Growth of structure was steered by operant conditioning in one design at a time, on a single timescale. So there was no possibility of Baldwinian effects: adaptive processes could not interact across different timescales. Spatially, metal threads were mostly constrained to grow in two dimensions,[12] and could not grow outside a solution full of metal ions. The system therefore could not influence the world beyond modulating the waveform of its electrical output. The configuration space (the set of all possible configurations the structure could perform) was thus well-specified by whatever displays or motors Pask happened to attach to its outputs.

In order to achieve epistemic autonomy, the system would also need to take control over the kinds of motor organs, effectors, and configurations it has available to influence the world [36]. It is contended here that autonomy can evolve by degrees with increasingly protean machines. This follows from the fact that a morphologically-static machine is bounded, if not fully determined by the complete specification of its parts—whereas a more or less protean machine more or less determines its own relationship with the world (and with its designer).

For this to be achieved, however, several conceptual and operational challenges will need to be overcome. Any adaptive protean machine will likely need to balance an energy budget, among many other concerns, and metamorphosis is metabolically expensive—for animals, at least. This is why caterpillars are such voracious feeders: they need to store up enough energy to change their body all at once into a butterfly. The question is thus to what extent are relatively static periods of development merely a proximate detail of animals, rather than an ultimate mechanism [234] or law of

[12]Given that electrodeposition is also possible in three dimensions [145], it should also be possible to grow Paskian sensors out of plane.

adaptive success. It is unclear when and where (if at all) a robot's morphological freedom should likewise be clamped (e.g. as a crawling eating machine) in order to accelerate the acquisition of new adaptive behaviors (e.g. land-to-air transition).

Appreciable morphological change in animals is not only expensive, it is often unnecessary. A flying organism or robot that is passively gliding through the air ought not shapeshift, especially out of wings. There would seem to be a kind of ecological speed limit for restructuring a body plan relative to changes in the world around it. Or perhaps there are tricks to perceptual and motoric constancy during high speed morphological change that we simply cannot imagine.

In either case, assuming that an engineered system is capable of adaptive morphological change in response to environmental conditions, it is unclear how it should do so, beyond the examples of morphological plasticity observed in nature. Examples include Wolff's law [198]—bone grows in response to particular mechanical loading profiles—and Davis' law—soft tissue increases in strength in response to intermittent mechanical demands. One can envisage other such laws that are not known to occur in biology but could be helpful in a specific artificial system, such as liquefying in response to pressure, or increasing stickiness in response to radiation [168].

If protean machines are eventually to perform useful work, these tricks and mechanisms of developmental plasticity must not only be understood but optimized. Optimizing a protean system that may continuously vary its structure, shape and material properties is extremely nonintuitive and underexplored. Thus, a study of the adaptive properties of such systems—and how they can best be optimized to render useful work—is documented here.

1.13 Overview of the Thesis

This thesis is organized around six published papers, which form chapters 2 to 7. The papers are presented chronologically with the exception of chapter 2, which was written last but is presented first. Chapter 2 contains a fairly standard Lipsonian experiment in which robot structure was permitted to vary in evolutionary time T, but not developmental time t; evolved designs transferred to reality with static structures. Succeeding chapters introduce experiments in which increasingly more aspects of the robot's design are allowed to vary in different combinations in T and/or t. Chapter 3 explores shape change in both T and t. Chapter 4 allows configuration-trajectories to vary alongside shape, in both T and t. Chapter 5 permits changes in structure and configuration-trajectories in T, as well as material change in both T and t. Chapter 6 investigates change to configuration trajectories in $T < s$, structural change at $T = s$, and then shape change for recovery in $T > s$; evolved designs were transferred to a physical robot that could vary both shape and configuration. The penultimate chapter

describes computer-designed organisms whose structure and material were evolved in silico under configuration variance in T; and whose structure, shape, material and configuration, all vary during development in vivo. The final chapter summarizes this progression, and, under the assumption that the reader is familiar with the main results in the intervening chapters, states the contributions in greater detail than was possible in section 1.2.

Chapter 2

Structure

Appeared as:
S. Kriegman et al., Scalable sim-to-real transfer of soft robot designs. In *Proceedings of the IEEE International Conference on Soft Robotics (RoboSoft)* (2020).

Abstract:
The manual design of soft robots and their controllers is notoriously challenging, but it could be augmented—or, in some cases, entirely replaced—by automated design tools. Machine learning algorithms can automatically propose, test, and refine designs in simulation, and the most promising ones can then be manufactured in reality (sim2real). However, it is currently not known how to guarantee that behavior generated in simulation can be preserved when deployed in reality. Although many previous studies have devised training protocols that facilitate sim2real transfer of control polices, little to no work has investigated the simulation-reality gap as a function of morphology. This is due in part to an overall lack of tools capable of systematically designing and rapidly manufacturing robots. Here we introduce a low cost, open source, and modular soft robot design and construction kit, and use it to simulate, fabricate, and measure the simulation-reality gap of minimally complex yet soft, locomoting machines. We prove the scalability of this approach by transferring an order of magnitude more robot designs from simulation to reality than any other method. The kit and its instructions can be found here: voxcraft.github.io

2.1 Introduction

The simulation-reality gap[1] for rigid-bodied robots is steadily closing. Computational models of rigid body dynamics can now be regularized and tuned so that control policies optimized in simulation are just as successful when tested on the physical system [27, 104]. The reality gap for soft robots, on the other hand, remains uncharted. It could be wider than the gap for rigid bodies, or not. Soft bodies are more challenging

[1]Henceforth, "the reality gap"—as coined by Jakobi et al. [108].

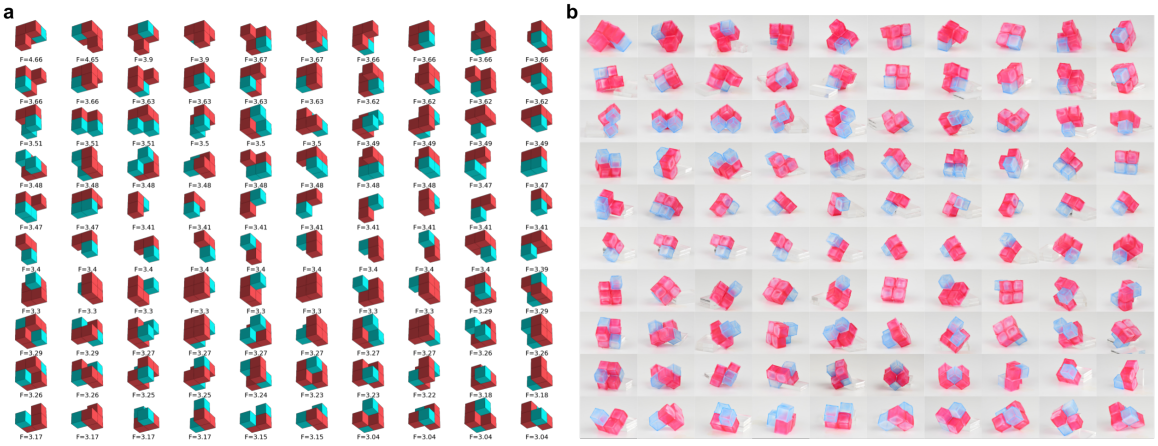


Figure 2.1: The top 100 simulated 2-by-2-by-2 configurations of passive (cyan) and volumetrically-actuating (red) voxels (**a**) were manufactured in reality (**b**).

to accurately simulate, design, and precisely control. But, they are also, by definition, more permissive to simulation inaccuracies, design flaws, and control precision: A soft gripper or foot will passively conform to complex objects and terrain, reducing the burden on the simulator to perfectly capture any single, "true" surface contact geometry.

Quantifying which soft robot designs, policies and behaviors can be faithfully simulated is critical not only for robotics, but also synthetic approaches to understand functional plasticity of biological systems during development and regeneration. For both domains, testing candidate hypotheses in reality is expensive, time consuming, and, in some cases, dangerous. With the recent development of several high-space, many-body, GPU-accelerated soft body simulators [97, 144], sim2real for soft robotics and synthetic biology has become more feasible. However, because these simulators have yet to be employed to design physical systems, their transferability is currently unknown.

Previous work has demonstrated methods that promote successful sim2real transferal of soft object manipulation but not soft robot behavior. For example, a rigid-bodied robot arm was successfully trained in simulation to fold towels and drape pieces of cloth over a hanger [149]. However, the reality gap was not quantified beyond a binary success rate for each task. Additionally, the robot's geometry was fixed and controllers were then optimized for it, whereas in the work reported here, the robot's geometry is part of the solution space.

Hiller and Lipson [94] evolved the overall geometry and distribution of hard and soft materials in simulation, and transferred the structures and passive dynamics of various cantilever beams. In a separate experiment that included actuating materials, Hiller and Lipson evolved the morphology and behavior of soft robots in simulation,

Table 2.1: Summary of published sim2real transference.

Author/citation	*Year*	*Controllers*	*Morphologies*
Miglino et al. [152]	1994	1	1
Jakobi et al. [108]	1995	2	1
Harvey et al. [89]	1997	4	1
Lipson and Pollack [140]	2000	3	3
Bongard et al. [27]	2006	34	2
Hiller and Lipson [94]	2012	1	5
Koos et al. [116]	2012	2	2
Moeckel et al. [158]	2013	1	1
Caluwaerts et al. [35]	2014	2	1
Cully et al. [55]	2015	10	10
Cellucci et al. [38]	2017	1	3
Tobin et al. [235]	2017	1	1
Rusu et al. [200]	2017	1	1
Peng et al. [183]	2018	1	1
Pinto et al. [189]	2018	3	1
Tan et al. [230]	2018	2	1
Golemo et al. [84]	2018	1	1
Matas et al. [149]	2018	3	1
Kwiatkowski and Lipson [127]	2019	2	2
Hwangbo et al. [104]	2019	3	1
Kriegman et al. [124]	2019	1	5
Nachum et al. [166]	2019	3	1
Akkaya et al. [3]	2019	1	1
Rosser et al. [197]	2019	1	16
The results presented here	**2019**	**1**	**108**

and then built one of the evolved designs physically. However, in order to transfer the simulated behavior of this one design, the physical robot needed to be placed in a pressure and vacuum chamber, whereas the hundreds of soft robot designs built here can be internally pressurized and actuated.

More recently, Kriegman et al. [124] subjected a simulated soft robot (composed of elastic voxels) to a series of damage scenarios that removed increasingly more of the robot's structure. In each scenario, the robot was challenged to recover function (locomotion) by deforming its remnant structure, without changing its predamage control policy. A pair of recovery strategies, automatically discovered by an evolutionary algorithm in simulation, were transferred to reality (using silicone "voxels"), but function was not: The physical system could deform its resting structure as dictated by the recovery strategy, but it could not locomote, before or after damage. The physical robot was heavy, had high friction feet, and was symmetrically actuated in phase, so it just oscillated in place.

To determine the particular challenges and opportunities of soft robot transferal, it

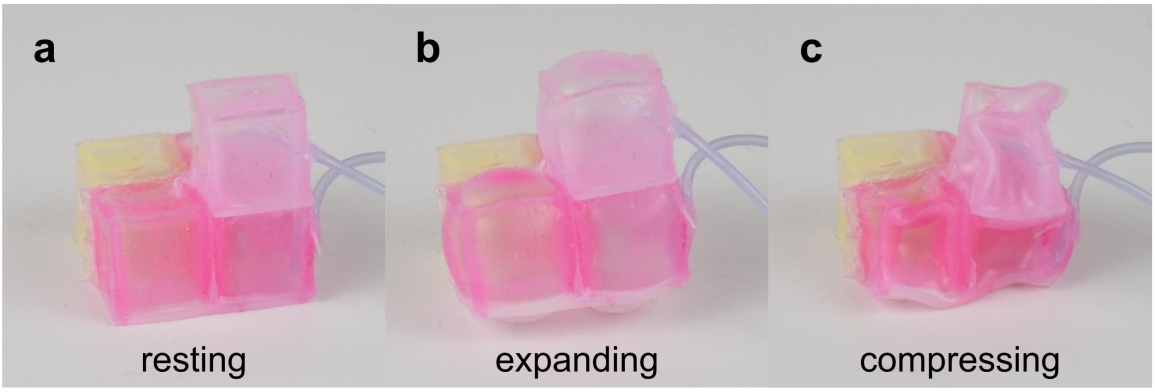


Figure 2.2: A random morphology in the design space shown at atmospheric (resting; **a**), positive (expanding; **b**), and negative (compressing; **c**) pressure.

would be useful to greatly scale up the number of design/policy pairs transferred. To this end, we present a soft robot design and construction kit based on the silicone voxel modules used in [124], but miniaturized to increase stability, simplified to improve reproducibility, and arbitrarily actuated to permit the transferal of locomotion.

Other modular yet rigid-bodied robot design and construction kits exist, such as Molecubes [258]. However, our kit is easier, faster, cheaper, and safer to use. In short, silicone is molded into hollow voxels, and tubing is attached to supply low pressure actuation from a hand pump, causing volumetric changes in one or more of the voxels (Figs. 2.2 and 2.3). For simple behaviors robust to actuation noise, there is no need to use a highly-pressurized air supply or program microcontrollers for open-loop control. There are also no expensive motors or power supplies.

Here, we employ the kit as an instrument to measure the reality gap as a function of morphology (Table 2.1). To do so, we fabricated 108 morphologies (transferal of structure) and compared the behavior of nine simulated designs to their silicone equivalents (transferal of behavior). We hope that the kit's affordability, safety, speed, and simplicity will generate increasingly more, and more reproducible, data about the automated design of increasingly competent soft machines.

2.2 Methods

The design space.

Following [94] and [124], our kit uses elastic voxels as building blocks of structure. Here, we considered a 2-by-2-by-2 cartesian lattice workspace, within which voxels were connected together to form a robot. At each x,y,z coordinate, voxels could either be passive, volumetrically actuated, or absent, yielding a total of $3^8 = 6561$ different

configurations. We evaluated each configuration in simulation.

The simulation.

We used the soft-body physics engine Voxelyze [95] to simulate robots composed of actuating and/or passive, elastic voxels. The simulator models the distance between adjacent voxels as Euler-Bernoulli beams (critically damped; $\zeta = 1$). Additionally, a collision detection system monitors the distance between the voxels on the surface of the robot at each timestep. If a pair of surface voxels are detected to collide (intersect), a temporary beam (underdamped; $\zeta = 0.8$) is constructed between the two until the collision is resolved.

Designs were simulated with a gravitational acceleration of -9.81 m/s^2, and initialized on top of an infinite surface plane at $z = 0$. Coulomb friction is applied to voxels in contact with the surface plane. Voxels were simulated to have 1 cm^3 resting volume (resting beam lengths), with Young's modulus 10^7 Pa, Poisson's ratio 0.35, and coefficients of static and kinetic friction of 1 and 0.5, respectively. These hyperparameters were adopted from [124]. For more details about how the physics are actually modeled, see [95].

Volumetric actuation was implemented by varying the rest length between voxels, in all three dimensions, when computing the elastic force between them. Volumetric expansion in simulation and reality are both roughly spherical (Fig. 2.2b), but compression in reality is more complex and difficult to simulate: the voxels buckle (Fig. 2.2c). So volumetric actuation was here limited to expansion only (+90% rest volume). The active voxels expand in phase with each other as dictated by a central pattern generator: a sine wave with frequency 4 Hz and amplitude 1.9 cm^3. When the sine wave is at or below zero, the active voxels remain at their resting volume (1 cm^3). This produced quasistatic dynamics.

Each design was simulated for 8831 timesteps, with a stepsize of 0.000453 seconds, resulting in a total simulation time of 4 seconds. During the first 552 times steps (0.25 sec), the design was allowed to settle under gravity before actuation begins, ensuring that movement (if any) is a result of actuation, rather than passively falling forward. Just before actuation, the design's initial center of mass is recorded as (x_0, y_0, z_0). The active voxels are then actuated for 3.75 sec at 4 Hz, or 15 actuation cycles.

An exhaustive search of all 6561 designs (in batches of 50) took 58 CPU hours (1.8 wall-clock hours) on a single AMD Ryzen threadripper 1950X 16-core/32-thread processor. Fitness was taken to be the net displacement (away from the origin in any direction) of the design's center of mass, in terms of euclidean distance in the plane, where the origin is defined by the x, y components of the design's initial center of

mass (x_0, y_0). Fitness is thus defined as:

$$F = \sqrt{(x_t - x_0)^2 + (y_t - y_0)^2}\,, \tag{2.1}$$

where x_t, y_t are the final coordinates of the design at the end of the evaluation period.

Reality.

Following Kriegman et al. [124], simulated voxels were realized physically as pneumatically actuated, hollow silicone voxels. The physical robot in [124] was constructed to transfer symmetrical shape change, so its actuated voxels were distributed symmetrically and hooked into a single pressure inlet. Thus, pressure oscillations occurred symmetrically in phase, and the robot could only pulse in place. Moreover, due to thin voxel walls relative to overall voxel size, and the tubing and glue used to bond them together, the robot in [124] could not fully support its own weight. The robot was lifted off the ground by placing it on top of a small petri dish, positioned underneath a segment of entirely passive voxels in the center of the robot's ventral surface. This permitted ventral (and more extreme global) changes in surface curvature, yielding successful sim2real transfer of shape change, but not locomotion.

The construction kit presented here rectifies the weight issue by miniaturizing the voxels—voxel length was halved (from 3cm to 1.5cm) and the wall thickness remained the same (1mm), reducing voxel mass from 4.3g to 1.2g (including tubing but not pneumatic connectors). Further, the inter-voxel tubing and glue was replaced with holes punched through the walls of adjacent active voxels in the same x,y slice, before attaching them with a shared bottom layer (Fig. 2.3d). Finally, locomotion is now possible because separate contiguous sections of voxels in each slice can be arbitrarily actuated in or out of phase with other sections across the body.

The build protocol.

The voxels were manufactured using a single-axis rotational molding machine. First, an open-face mold was fabricated by interlacing 26 acrylic strips into a flat base, to form a lattice of cubic concavities, resembling an ice-cube tray (Fig. 2.3a). Mold components were laser-cut (VLS2.30, Universal Laser System) from a flat acrylic sheet with a thickness of 0.025 inch. Next, silicone (Dragon Skin 10 Fast; Smooth-On, Inc.) was poured into the acrylic mold (Fig. 2.3a), and a spatula was used to spread the silicone along the interior walls of each cavity (Fig. 2.3b). Colored pigment was added to each batch of silicone to indicate whether the voxel was active or passive, simplifying the assembly process. Here we used pink for active voxels and blue or yellow for passive voxels.

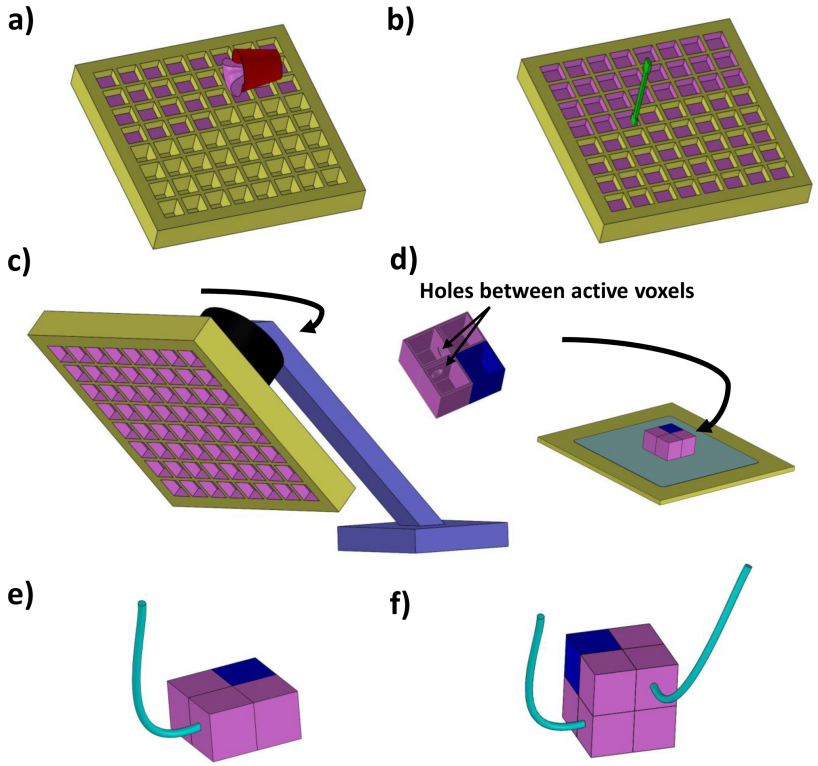


Figure 2.3: **Manufacturing modular soft robots.** Hollow, silicone voxels were created by partially filling an open-face mold with silicone (**a**), using a spatula to spread it along the interior walls (**b**), and then securing the mold to a 1-axis rotational molding machine (**c**). This process allowed excess silicone to drip out of the mold, while spreading the remaining silicone into a thin uniform layer. The cured, bottomless voxels were then appropriately arranged and connected for each x,y slice of the design, and bonded with a shared bottom layer (**d**). Finally, tubing was attached (**e**), and the slices were stacked and bonded to form the design (**f**). Video: **youtu.be/jbQ2T7jIYRU**.

The mold was then flipped upside down and secured to a 1-axis rotational molding machine. The machine was clamped to a table with binder clips, angled 45° relative to horizontal, and set to rotate 90° every 45 seconds (Fig. 2.3c). This allowed the silicone to flow and evenly coat the walls of the mold, as excess silicone dripped out. After the voxels partially cured for 25 minutes at room temperature, the mold was moved to an incubator, with a temperature of 60°C for another 20 minutes. (Without an incubator, the silicone will take 75 minutes to fully cure at room temperature.)

The above steps were then repeated to add an additional layer of silicone. Once the second layer cured, the bottomless voxels were removed from the mold using an X-Acto knife, and excess silicone around their edges was trimmed.

In the next step, each x,y slice (or dorsal plane) of the design was assembled by using Sil-Poxy (Smooth-On, Inc.) to bond adjacent voxels and prevent the slice from

shifting. Holes were then punched between adjacent active voxels so that contiguous collections of voxels could be actuated together in phase. Each actuator group needed to contain at least one voxel on the surface of the design so that it could be controlled by an external pressure inlet. To create the bottom layer, two 1mm-thick rulers were attached to an acrylic substrate using double-sided tape and silicone was poured in the space between them. Then, the slice of bottomless voxels was flipped, open-side down, onto this uncured silicone layer (Fig. 2.3d).

After the bottom layer cured, a thin layer of silicone was applied with a popsicle stick along the outermost portions of the interstices of the voxels, bonding adjacent voxels (without gluing over inter-voxel holes). Then, the slice was cut from the silicone sheet and a hole was poked into the side of one exterior voxel from each group of active voxels. Next, a 1/32" ID silicone tube was inserted into the hole, and glued in place with Sil-Poxy, applied with a Q-tip (Fig. 2.3e). The end of this tube was then connected to a straight pneumatic connector, which was connected to 1/16" ID silicone tubing.

Occasional imperfections in alignment, silicone thickness, or inter-voxel hole sizes would result in leaky structures. Leaks were detected by filling a beaker with water, submerging the voxels, and inflating them. Bubbles would emanate from leaks, which were repaired with Sil-Poxy. After repairing any leaks, the slices were stacked on top of each other and bonded together using a thin layer of silicone (Fig. 2.3e). Finally, these layers were connected pneumatically with assorted pneumatic connectors, attached to 1/16" ID silicone tubes.

2.3 Results

To test the effects of morphology on fitness and sim2real transfer success, it is useful to first visualize the design space. However, because there are eight cartesian voxel coordinates in the chosen workspace, the design space here is eight dimensional, which is difficult to draw (let alone conceptualize) without dimensionality reduction. By nesting the dimensions of a search space onto a single plot (Fig. 2.4), the entire space can be visualized as a 2D heatmap. This strategy was used by Cully et al. [55] to neatly visualize the predicted fitness of a very large library of control policies, as a

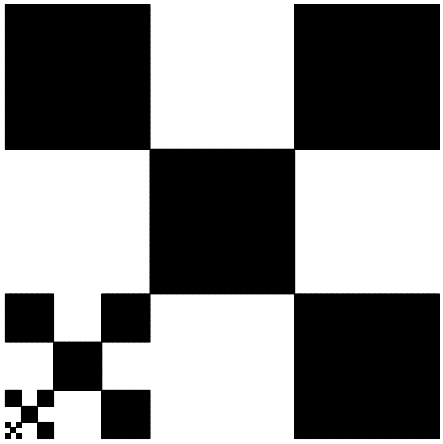


Figure 2.4: The 2D tessellation of 8D ternary vector space used in Fig. 2.5.

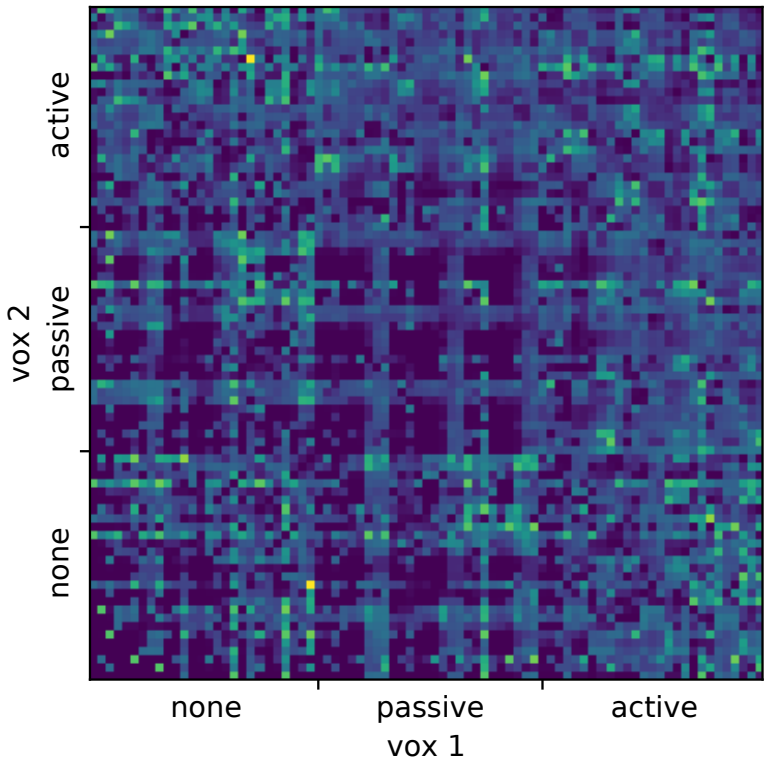


Figure 2.5: **Simulating modular soft robots.** The design space is plotted as a heatmap, containing one cell for each of the 6561 possible configurations. Lighter colored cells are fitter designs (Eq. 2.1). Each design is defined by a vector of eight ternary values, indicating what kind of voxel (none, passive, or active) the design contains at the eight lattice points in the $2 \times 2 \times 2$ workspace. The 8D ternary vector is reduced to a 2D heatmap by nesting pairs of dimensions within each other: four, nested 3×3 grids result in a $3^4 \times 3^4 = 81 \times 81$ overall heatmap.

function of the time a robot's six limbs were in contact with the simulated ground plane: 6D quinary control space was mapped to 2D, by nesting pairs of dimensions within each other.

Here, the 8D ternary morphology space was reduced to 2D by plotting pairs of dimensions nested within each other (Fig. 2.5). The pixel in the exact center of Fig. 2.5, for instance, represents the configuration consisting entirely of passive voxels, and thus cannot locomote ($F = 0$). Likewise, the pixel in the top right-hand corner of the heatmap represents the configuration of all active voxels (Fig. 2.6d), which actuated symmetrically in phase, and thus (given its flat ventral surface) could not locomote across the flat ground plane ($F = 0$). Finally, the pixel in the bottom left-hand corner contains no voxels at all, and thus $F = 0$.

For locomotion, a good design obviously needs to have a body, rather than none

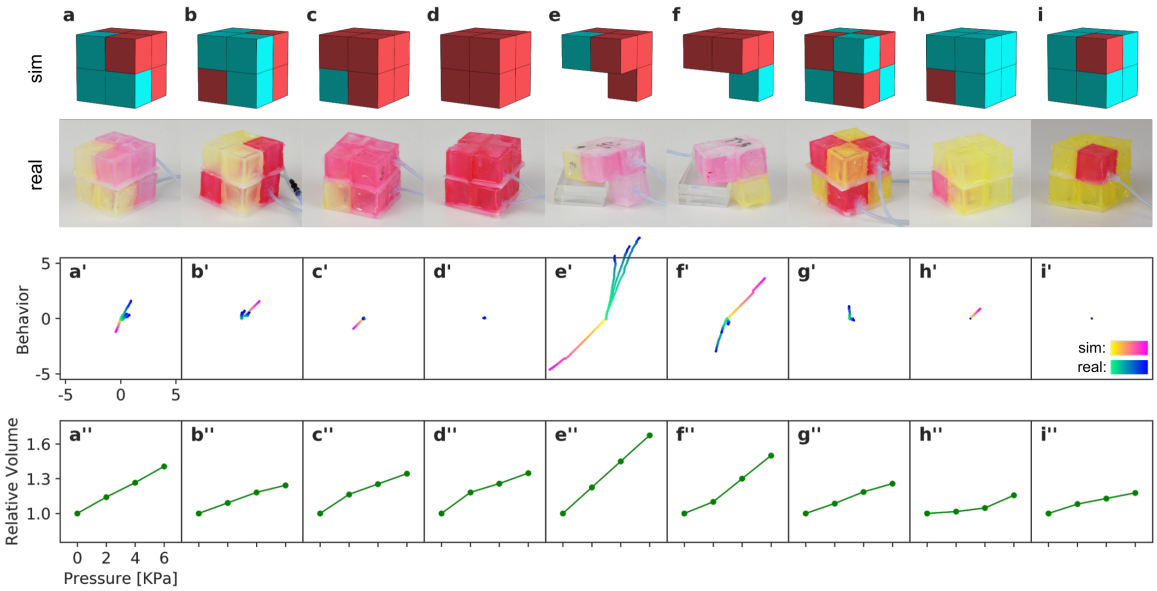


Figure 2.6: **Measuring transferal from simulation to reality**. Nine designs (**a-i**) were evaluated three times each in reality (green-to-blue gradient colored curves in **a'-i'**). The behavioral trajectories start at the origin (green) and end at the robot's final XY destination (blue) (in centimeters). The simulated movement tracks (yellow-to-pink curves) are superimposed on top of the real ones. The relative volume (normalized by rest volume) was also recorded for each design at four points during actuation under water (**a''-i''**). The simulated and real behavior of designs e and f can be observed here: **youtu.be/UqjvmkYa9u4**.

at all. With open-loop, in-phase actuation, designs also need to have asymmetrical mass and/or actuator distributions, or they will not generate any forward movement. However it is not clear, even for this minimal design space, exactly which asymmetrical designs will yield the highest fitness. Yet we can see small clusters and lines of similarly colored pixels in Fig. 2.5, representing morphologically similar designs with similar fitness. This suggests that these configurations and substructures would be relatively stable under random mutations or errors in fabrication.

Because fitness was measured by displacement in any direction away from the origin (Eq. 2.1), there are four configurations—rotations, in the x,y plane, of a single geometry and distribution of passive and active voxels—with different behaviors (they moved in different directions) but very similar (if not identical) fitness. There were also some configurations that, when rotated upward (in the x,z or y,z plane) fell into the same basic orientation and behavior but with a slightly different heading. Thus, configurations with similar fitness (similarly colored pixels) are reflected across multiple, nested planes of symmetry in Fig. 2.5. These symmetries can also be seen in the manufactured robots (Fig. 2.1b). The uniqueness of designs (i.e., the size of the search space of morphologies) is therefore a function of how behavior is measured.

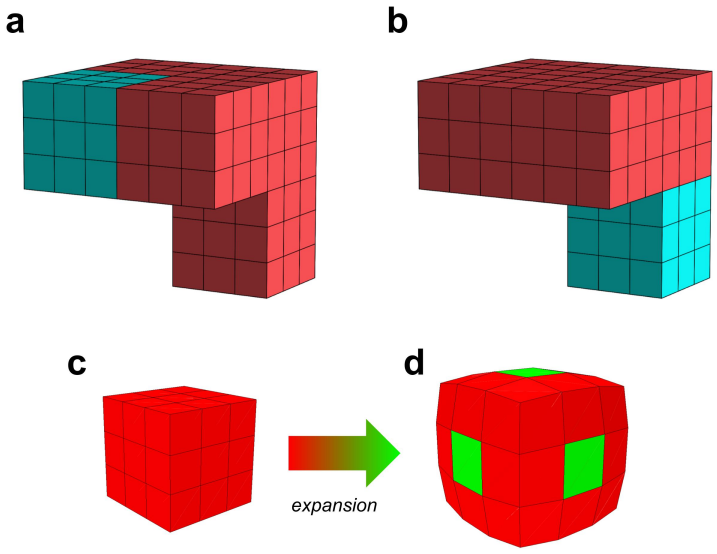


Figure 2.7: A higher resolution model in which each silicone voxel is approximated by a 3-by-3-by-3 group of simulated subvoxels: a high-res voxel. The design in **a** and **b** are high-res instantiations of those in Figs. 2.6e and 2.6f, respectively. Spherical volumetric expansion in a high-res voxel (**c**) was approximated by increasing the rest length between the centermost subvoxel and the subvoxels at center of each face (green subvoxels in **d**).

Fig. 2.6 shows the behavior of nine different designs in simulation and reality. The real robot was actuated 90 times at 6 kPa pressure on a surface covered with cornstarch (Argo®, ACH Food Companies, Inc.) to reduce friction, and is compared to 23 simulated actuation cycles. Seven of the nine designs filled the cubic workspace with passive and active voxels, while the other two share a more complex geometry: a single-voxel limb attached to the face of a 2-by-2 plane of voxels (Fig. 2.6e,f). In one, the limb is active (Fig. 2.6e), in the other it is passive (Fig. 2.6f). These two designs achieved the two highest fitness scores (Eq. 2.1), in both simulation and reality.

By this measure, the reality gap appears small. However, these simulated designs move very differently from their manufactured equivalents. The simulated morphology in Fig. 2.6e pushes off its active limb, whereas in reality the design uses its limb to pull itself forward, in the opposite direction. Likewise, the simulated morphology in Fig. 2.6f pushes off its active 2-by-2 torso, whereas in reality the design uses its torso to pull itself forward, in the opposite direction.

Majidi et al. [147] showed that the interfacial shear strength and coefficient of friction of the surface on which their soft robot undulated determined the direction of locomotion. They decomposed friction into load- and area controlled terms for point and surface contacts, respectively. On slippery surfaces with low interfacial shear resistance, the robot anchored about the point contact (expanded section) for

locomotion and pulled its surface contact (passive segment). However, on surfaces with high interfacial shear resistance, the robot anchored about the surface contact and pulled the point contact toward it. We hypothesize that such differences in tribological properties could have caused our designs to move in opposite directions in simulation and reality.

In an attempt to test this hypothesis and reduce the simulation-reality discrepancies that cause the virtual configurations in Fig. 2.6 to move differently than their physical realizations, we performed a grid search of various simulation hyperparameters, including the coefficients of static and kinetic friction. However, we could not identify a pair of friction coefficients that resulted in correct movement heading for all nine of the behaving designs (Fig. 2.6a'-i'). This could be due to either low precision or low accuracy of the model. To isolate and test the former possibility, we increased the resolution of the simulated surface contact geometry by modeling each silicone voxel as a 3-by-3-by-3 group of simulated "subvoxels" (Fig. 2.7), and then re-ran the parameter sweep. Still, we could not find friction settings in which the simulated movement direction matched the ground truth across all designs simultaneously. This suggests that the accuracy of Coulomb friction model may be insufficient to model this type of movement.

The Coulomb approximation assumes that friction is simply proportional to the vertical component N of the reaction force, and independent of the contact area. However, friction is also a function of the surface area and interfacial shear strength τ, a fixed constant which is mostly governed by adhesion or mechanical interlocking between the contacting surfaces. A better model would thus consider friction as a function of both the normal force and the interfacial shear strength. However, before fundamentally changing the simulator, we plan to evaluate designs in noisy environments with imperfect control over actuation characteristics to avoid ascribing high fitness to designs that exploited unrealistic properties of the simulation [108]. Additionally, data from reality could be used to automatically tune the geometry and resolution of the simulated finite elements [27], or to predict the kinds of behaviors that are more likely to successfully transfer [116], and which should be tested next [27]. Concurrently, we are investigating additional physical surfaces with varied tribological properties in an attempt to match reality to simulation.

2.4 Discussion

In this paper, we introduced a low cost, open source platform for designing and rapidly building soft robots, and used it to transfer 108 different morphologies (voxels on a cartesian grid) from simulation to reality. We then measured the reality gap as function of the robot's design (geometry and distribution of passive and actuating

voxels) by tracking the behavior of nine transferred morphologies. Under one measure (net displacement) the reality gap appeared rather small, but under another (velocity) the gap was much wider, likely due to oversimplified tribological contacts between the simulated ground plane and the robot's ventral surface [147].

Although most of the transferred designs (99 out of the 108) were not actuated in reality, they nevertheless served an important function: they were *sketches*. Sketches let us think more clearly about the behavior or properties (e.g., stability) of a design without investing the additional time and resources required to fully build and examine the design itself. Sketches, in other words, greatly increase the breadth of exploration in design space. All sim2real methods embrace this utility of simplifying sketches. Simulation, after all, is also a sketch.

However, there is a tacit assumption in robotics about Depth First. A typical sim2real experiment begins by sending a complicated robot design across the reality gap, and then endeavors to learn transferable policies that control the morphology in all its complexity. But this is not how most design proceeds. An architect first roughly sketches a structure, say, a bridge, on the back of a napkin. A diversity of designs are then generated, tweaked, discarded or provisionally accepted—at this shallow level of napkin realism—before more detailed blueprints are drawn under more stringent constraints. Blueprints, too, undergo this breadthwise evolution, before the most promising are realized physically, first as scale models (built from matchsticks and glue instead of concrete and steel), then, finally, at full scale and cost. This incrementally weeds out nontransferable features and adds mechanical complexity only when and where it is necessary to do so, rather than globally from the outset.

The assumption that the reality gap can be bridged by policy search alone, with a single robot design, is groundless. The desired behavior of a robot is typically much more complicated than that of architecture. This suggests the necessity of more, not less, sketches. Soft robots are more complicated still. This makes their automated design that much more appealing, but implies the need for even greater breadth of sketches, at more intermediate levels of realism. Though not every experiment will need to start from a blank slate. Instead, designers (whether human or AI) could leverage prior knowledge to reject truly awful designs before sketching them in the first place. The designs transferred here add to a growing database (prior probabilities) about which and how well different soft robot designs and behaviors can be realized physically. Our construction kit has the potential to further increase this data by lowering not only cost and build times but also the barrier of entry to soft robotics for non-experts.

The generality of such data beyond robotics is currently not known, but it could also have important implications for developmental biology and regenerative medicine. The bioelectric and genetic control policies that orchestrate adaptive remodeling of growth and form in organisms are not yet understood, but could, in future, be reverse-

engineered by machine learning, and then controlled externally to induce regeneration in otherwise non-regenerative organisms (such as adult humans), or to reprogram otherwise unbounded cancerous growth toward functional organogenesis [134]. However, such advances in regenerative medicine and synthetic morphology will only be possible if hypotheses generated in simulation are transferable and testable in reality.

Chapter 3

Shape

Appeared as:
S. Kriegman et al., A minimal developmental model can increase evolvability in soft robots. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO)* (2017).

Abstract:
Different subsystems of organisms adapt over many time scales, such as rapid changes in the nervous system (learning), slower morphological and neurological change over the lifetime of the organism (postnatal development), and change over many generations (evolution). Much work has focused on instantiating learning or evolution in robots, but relatively little on development. Although many theories have been forwarded as to how development can aid evolution, it is difficult to isolate each such proposed mechanism. Thus, here we introduce a minimal yet embodied model of development: the body of the robot changes over its lifetime, yet growth is not influenced by the environment. We show that even this simple developmental model confers evolvability because it allows evolution to sweep over a larger range of body plans than an equivalent non-developmental system, and subsequent heterochronic mutations 'lock in' this body plan in more morphologically-static descendants. Future work will involve gradually complexifying the developmental model to determine when and how such added complexity increases evolvability.

3.1 Introduction

Many theories have been proposed as to how development can confer evolvability. Selfish gene theory [57] suggests that prenatal development from a single-celled egg is not a superfluous byproduct of evolution, but is instead a critical process that ensures uniformity among genes contained within a single organism and in turn their cooperation towards mutual reproduction. Developmental plasticity, the ability of an organism to modify its form in response to environmental conditions, is believed to

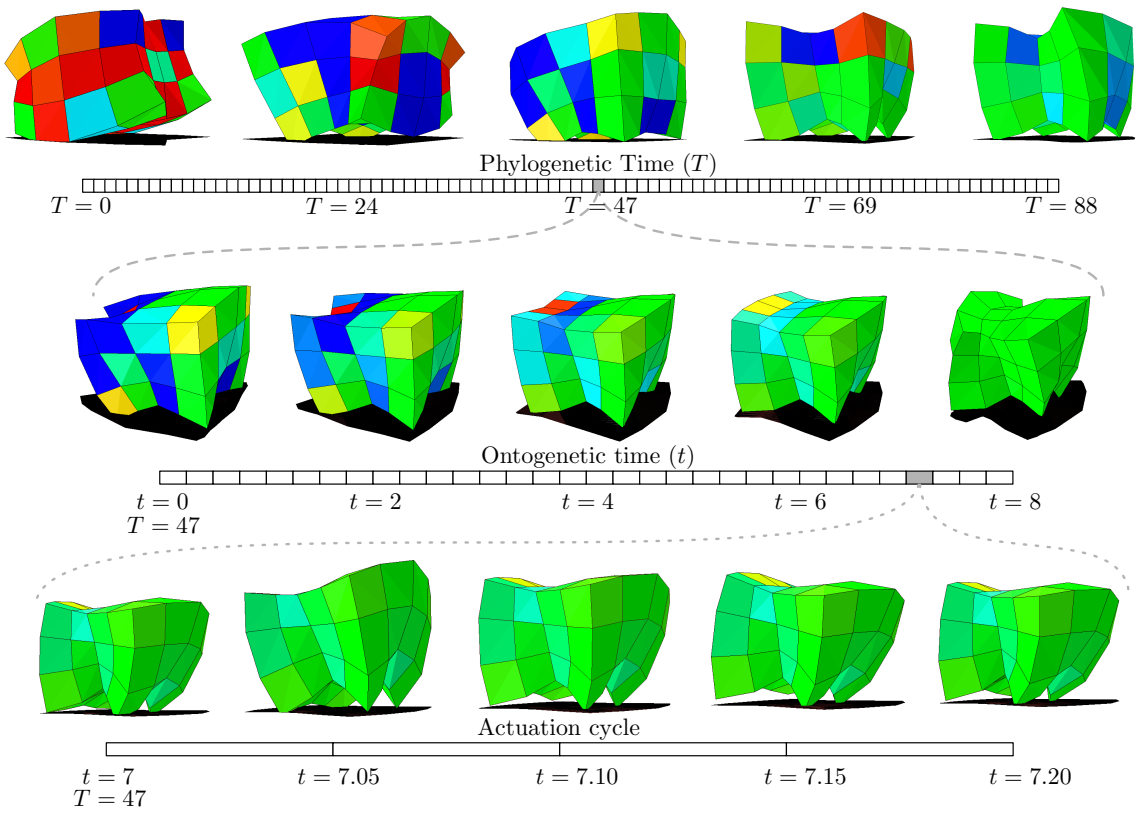


Figure 3.1: **The evolutionary history of an Evo-Devo robot.** One of the five phylogenies is broken down into five ontogenies which is in turn shown at five points in its actuation cycle. Voxel color indicates the remaining development. Blue for shrinkage, red for growing, and green for no further change. This robot is featured in video at youtu.be/gXf2Chu4L9A

play a crucial role in the origin and diversification of novel traits [157]. Others have shown that development can in effect 'encode', and thus avoid on a much shorter time scale, constraints that would otherwise be encountered and suffered by non-developmental systems [119].

Several models that specifically address development of embodied agents have been reported in the literature. For example Eggenberger [68] demonstrated how shape could emerge during growth in response to physical forces acting on the growing entity. Bongard [29] adopted models of genetic regulatory networks to demonstrate how evolution could shape the developmental trajectories of embodied agents. Later, it was shown how such development could lead to a form of self-scaffolding that smoothed the fitness landscape and thus increased evolvability [28]. Miller [154] introduced a developmental model that enabled growing organisms to regrow structure removed by damage or other environmental stress.

In the spirit of Beer's minimal cognition experiments [17], we introduce here a minimal model of morphological development in embodied agents (figure 3.2). This model strips away some aspects of other developmental models, such as those that reorganize the genotype to phenotype mapping [29, 68, 119] or allow the agent's environment to influence its development [96, 154]. We use soft robots as our model agents since they provide many more degrees of developmental freedom compared to rigid bodies, and can in principle reduce human designer bias. Here, development is monotonic and irreversible, predetermined by genetic code without any sensory feedback from the environment, and is thus *ballistic* in nature rather than adaptive.

While biological development occurs along a time axis, it has been implied in some developmental models that time provides only an avenue for regularities to form across space, and that only the resulting fixed form — its spatial patterns, repetition and symmetry — are necessary for increasing evolvability. Compositional pattern producing networks (CPPNs, [221]) explicitly make this assumption in their abstraction of development which collapses the time line to a single point. While CPPNs have proven to be an invaluable resource in evolutionary robotics [42], we argue here that discarding time may in some cases reduce evolvability and that there exist fundamental benefits of time itself in evolving systems.

In this paper, we examine two distinct ways by which ballistic development can increase evolvability. First, we show how an ontogenetic time scale provides evolution with a simple mechanism for inducing mutations with a *range* of magnitude of phenotypic impact: mutations that occur early in the life time of an agent have relatively large effects while those that occur later have smaller effects. This is important since, according to Fisher's geometric model [76], the likelihood a mutation is beneficial is inversely proportional to its magnitude: Small mutations are less likely to break an *existing* solution. Larger exploratory mutations, although less likely to be beneficial on average, are more likely to provide an occasional path out of local optima. Second, we posit that changing ontogenies diversify targets for natural selection to act upon, and that advantageous traits 'discovered' by the phenotype during this change can become subject to heritable modification through the 'Baldwin Effect' [66].

Hinton and Nowlan [96] relied on this second effect when they demonstrated how learning could guide evolution towards a solution to which no evolutionary path led. We consider a similar hypothesis with embodied robots and ballistic development, rather than a disembodied bitstring and random search. We demonstrate how open-loop morphological development, without feedback from the environment and without direct communication to the genotype, can similarly alter the search space in which evolution operates making search much easier. Hinton & Nowlan's model of learning was a type of environment-mediated development, in the sense that developmental change stops when the 'correct specification' is found, and this information is then used to bias selection towards individuals that find the solution more quickly. Our

work demonstrates that this explicit suppression of development is not necessary; and that completely undirected morphological change is enough to confer evolvability.

3.2 Methods

All experiments[1] were performed in the open-source soft-body physics simulator *Voxelyze*, which is described in detail in Hiller and Lipson [95].

We consider a locomotion task for soft robots composed of a $4 \times 4 \times 3$ grid of voxels (see figure 3.1 for example). Each voxel within and across robots is identical with one exception: its volume. At any given time, a robot is completely specified by an array of *resting volumes*, one for each of its 48 constituent voxels. If the resting volumes are static across time then a robot's genotype is this array of 48 voxel volumes; however, because we enforce bilateral symmetry, a genome of half that size is sufficient. On top of the deformation imposed by the genome, each voxel is volumetrically actuated according to a global signal that varies sinusoidally in volume over time (figure 3.2). The actuation is a linear contraction/expansion from their baseline resting volume.

Under this type of rhythmic actuation, many asymmetrical mass distributions will elicit locomotion to some extent. For instance, a simple design, with larger voxels in its front half relative to those in its back half, may be mobile when its voxels are actuated uniformly. Although this design would be rather inefficient since it most likely drags much of its body across the floor as it moves. More productive designs are not so intuitive, even with this fixed controller.

An individual is evaluated for 8 seconds, or 32 actuation cycles. The fitness was taken to be the distance, in the positive y direction, the robot's center of mass moved in 8 seconds, normalized by the robot's total volume. Thus, a robot with volume 48 that moves a distance of 48 will have the same fitness — a fitness of one — as a similarly shaped but smaller robot with volume 12 that moves a distance of 12. Distance here is measured in units that correspond to the length of a voxel with volume one. If, however, a robot rolls over onto its top layer of voxels it is assigned a fitness of zero and evaluation is terminated. This constraint prevents a rolling ball morphology from dominating more interesting gaits.

We have now built up all of necessary machinery of our first type of robot which we shall call the **Evo** robot. Populations of these robots can evolve: body plans change from generation to generation (phylogeny); but they can not develop: body plans maintain a fixed form, apart from actuation, while they behave within their lifetime (ontogeny).

[1] **https://github.com/skriegman/gecco-2017** contains the source code necessary for reproducing our results.

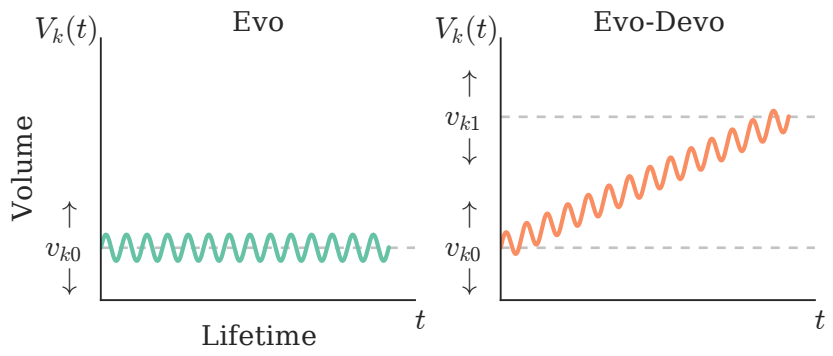


Figure 3.2: **The voxel picture.** The k^{th} voxel in an Evo robot maintains a fixed resting volume, v_{k0}, throughout the robot's lifetime. Sinusoidal actuation is applied on top of the resting volume. In contrast, the k^{th} voxel in an Evo-Devo robot changes linearly from a starting volume, v_{k0}, to a final volume, v_{k1}, over the robot's entire lifetime. Growth, the case when $v_{k1} > v_{k0}$, is pictured here, but shrinkage is also possible and occurs when $v_{k1} < v_{k0}$. When $v_{k1} = v_{k0}$, Evo-Devo voxels are behaviorally equivalent to Evo voxels. Voxels actuate at 4 Hz in our experiments (for 8 sec or 32 cycles) however actuation is drawn here at a lower frequency to better convey the sinusoidal volumetric structure in time.

We consider a second type of robot, the **Evo-Devo** robot, which inherits all of the properties of the Evo robot but has a special ability: Evo-Devo robots can develop as well as evolve. These robots are endowed with a *minimally complex* model of development in which resting volumes change linearly in ontogeny. We call this ballistic development to distinguish it from environment-mediated development. Ballistic development is monotonic with a fixed rate predetermined by a genetic program; its onset and termination are constrained at birth and death, respectively; it is strictly linear, without mid-course correction. The volume of the k^{th} voxel in an Evo-Devo robot changes linearly from a starting volume, v_{k0}, to a final volume, v_{k1}, within the lifetime of a robot (figure 3.2). Accordingly, the genotype of a robot that can develop is twice as large as that of robots that cannot develop, since there are two parameters (v_{k0} and v_{k1}) that determine the volume of the k^{th} voxel at any particular time. Although it is important to note that the space of possible morphologies (collection of resting volumes) is equivalent both with and without development.

From gene to volume.

Like most animals, our robots are bilaterally symmetrical. We build this constraint into our robots by constraining the 24 voxels on the positive x side of the robot to be equal to their counterparts on the other side of the y axis. Instead of 48 Evo genes, therefore, we end up with 24.

A single Evo gene stores the resting length, s_k, of the k^{th} voxel, which is cubed to

obtain the *resting* volume, $r_k(t)$, at any time, t, during the robot's lifetime.

$$r_k(t) = s_k^3 \qquad k = 1, 2, \ldots, 24 \tag{3.1}$$

The resting lengths may be any real value in the range (0.25, 1.75), inclusive. Note that the resting volume of an Evo robot does not depend on t, and is thus constant in ontogenetic time.

Volumetric actuation, $a(t)$ with amplitude u, and period w, takes the following general form in time.

$$a(t) = u * \sin(2\pi t/w) \tag{3.2}$$

Actuation is limited to $\pm 20\%$ and cycles four times per second ($u = 0.20$, $w = 0.25$ sec).

However, for smaller resting volumes, the actuation amplitude is limited and approaches zero (no actuation) as the resting volume goes to its lower bound, 0.25^3. This restriction is enforced to prevent opposite sides of a voxel from penetrating each other, effectively incurring negative volumes, which can lead to simulation instability. This dampening is applied only where $s_k < 1$ (shrinking voxels) and accomplished through the following function.

$$d(s_k) = \begin{cases} 1 & s_k \geq 1 \\ (4s_k - 1)/3 & s_k < 1 \end{cases} \tag{3.3}$$

Thus $d(s_k)$ is zero when $s_k = 0.25$, and is linearly increasing in $s_k \leq 1$. The true actuation, $\tilde{a}(t, s_k)$, is calculated by multiplying the unrestricted actuation, $a(t)$, by the limiting factor, $d(s_k)$.

$$\tilde{a}(t, s_k) = a(t) * d(s_k) \tag{3.4}$$

Actuation is then added to the resting volume to realize the *current* volume, $V_k(t)$, of the k^{th} voxel of an Evo robot at time t.

$$V_k(t) = [s_k + \tilde{a}(t, s_k)]^3 \tag{3.5}$$

For Evo-Devo robots, a gene is a pair of voxel lengths (s_{k0}, s_{k1}) corresponding to the k^{th} voxel's starting and final resting lengths, respectively. Thus, for a voxel in an Evo-Devo robot, the resting volume at time $t \in (0, \tau)$ is calculated as follows.

$$r_k(t) = \left[s_{k0} + \frac{t}{\tau}(s_{k1} - s_{k0}) \right]^3 \tag{3.6}$$

Where the difference in starting and final scale ($s_{k1} - s_{k0}$) determines the slope of linear development which may be positive (growth) or negative (shrinkage). The

current volume of the k^{th} voxel of an Evo-Devo robot is then determined by the following.

$$V_k(t) = \left[r_k^{1/3}(t) + \tilde{a}\left(t,\ r_k^{1/3}(t)\right)\right]^3 \tag{3.7}$$

Hence the starting resting volume, v_{k0}, and final resting volume, v_{k1}, are the current volumes at $t = 0$ and $t = \tau$, respectively.

$$\begin{aligned} v_{k0} &= V_k(0) = s_{k0}^3 \\ v_{k1} &= V_k(\tau) = s_{k1}^3 \end{aligned} \tag{3.8}$$

Note that an Evo gene is a special case of an Evo-Devo gene where $s_{k0} = s_{k1}$, or, equivalently, where $v_{k0} = v_{k1}$.

For convenience, let's define the current *total* volume of the robot across all 48 voxels as $Q(t)$.

$$Q(t) = 2\sum_{k=1}^{24} V_k(t) \tag{3.9}$$

We track the y position of the center of mass, $y(t)$, as well as the current total volume, $Q(t)$, at n discrete intervals within the lifetime of a robot. Fitness, F, is the sum of the distance traveled in time interval, divided by the average volume in the interval.

$$F = 2\sum_{t=1}^{n} \frac{y(t) - y(t-1)}{Q(t) + Q(t-1)} \tag{3.10}$$

We track $y(t)$ and $Q(t)$ 100 times per second. Since robots are evaluated for eight seconds, $n = 800$.

A direct encoding.

This paper differs from previous evolutionary robotics work that used Voxelyze [42–44] in that we evolve the volumes of a fixed collection of voxels, rather than the presence/absence of voxels in a bounding region. Another difference is that we do not employ the CPPN-NEAT evolutionary algorithm [221], but instead use a direct encoding with bilateral symmetry about the y axis. A comparison of encodings in our scenario is beyond the scope of this paper. However we noticed that the range of evolved morphologies here, under our particular settings, was much smaller than that of previous work which used voxels as building blocks, and that it is easier to reach extreme volumes for individual voxels using a direct encoding.

Apart from the difference in encoding, this work is by in large consistent with this previous work. We use the same physical environment as Cheney et al. [42]: a wide-open flat plain. The material properties of our voxels are also consistent with the 'muscle' voxel type from the palette in this work; although these voxels had a fixed resting volume of one ($s_k = 1$ for all k).

Evolutionary search.

We employ a standard evolutionary algorithm, Age-Fitness-Pareto Optimization (AFPO, [204]), which uses the concept of Pareto dominance and an objective of age (in addition to fitness) intended to promote diversity among candidate designs. For 30 runs, a population of 30 robots is evolved for 2000 generations. Every generation, the population is first doubled by creating modified copies of each individual in the population. Next, an additional random individual is injected into the population. Finally, selection reduces the population down to its original size according to the two objectives of fitness (maximized) and age (minimized).

The same number of parent voxels are mutated, on average, in both Evo and Evo-Devo children. Mutations follow a normal distribution ($\sigma = 0.75$) and are applied by first choosing what parameter types to mutate, and then choosing which voxels to mutate. For Evo robots, we simply visit each voxel (on the positive x side) of the parent and, with probability 0.5, mutate its single parameter value. For Evo-Devo parents, we flip a coin for each parameter to be mutated (if neither will be mutated, flip a final coin to choose one or the other). This results in a 25% chance of mutating both, and a 37.5% chance of mutating each of the two individual parameters alone. Then we apply the same mutation process as before in Evo robots: loop through each voxel of the parent and, with probability 0.5, mutate the selected parameter(s).

An artificially rugged landscape.

We did not fine-tune the mutation hyperparameters (scale and probability), but intentionally chose a relatively high probability of mutation in order to elicit a large mutational impact in an attempt to render evolutionary search more difficult. This removes easy to follow gradients in the search space — 'compressing' gentle slopes into abrupt cliffs — which make 'good designs' more difficult to find. Any one of these good solutions then, to a certain extent, become like Hinton & Nowlan's 'needle in a haystack' [96].

Note that there are other ways to enforce rugged fitness landscapes, and such landscapes are naturally occurring in many systems, though our particular task/environment is not one of them. Future work should investigate these tasks and environments with a fine-tuned mutation rate.

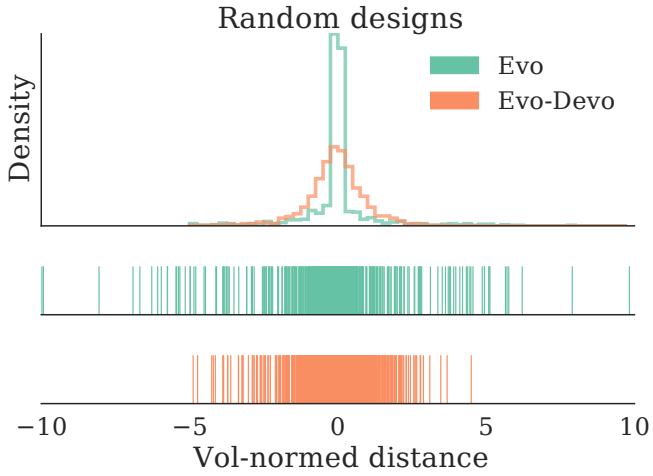


Figure 3.3: One thousand randomly generated robots for each group. The horizontal axes measure fitness: volume normalized distance in the positive y direction. The best overall designs are the best Evo robots since they maintain their good form as they behave. However, most designs are immobile (mode at zero) and Evo-Devo robots are more likely to move (less mass around zero) since they explore a continuum of body plans rather than a single static guess.

3.3 Results

In this section we present the results of our experiments[2] and indicate statistical significance under the Mann-Whitney U test where applicable.

Random search.

To get a sense of the evolutionary search space, prior to optimization, we randomly generated one thousand robots from each group (figure 3.3). The horizontal axes of figure 3.3 measure the fitness (equation 3.10) of our randomly generated designs. The top portion of this figure plots the histogram of relative frequencies, using equal bin sizes between groups. The mode is zero for both groups, meaning that the majority of designs are immobile.

The best possibility here is to randomly guess a good Evo robot since this good morphology is utilized for the full 32 actuation cycles. This is why the best random designs are Evo robots. However, the Evo-Devo distribution contains much less mass around zero than the Evo distribution. It follows that it is more likely that an Evo-Devo robot moves at all, if only temporarily, since this only requires some interval of the many morphologies it sweeps over to be mobile. Also note that while the total

[2] **https://youtu.be/gXf2Chu4L9A** directs to a video overview of our experiments.

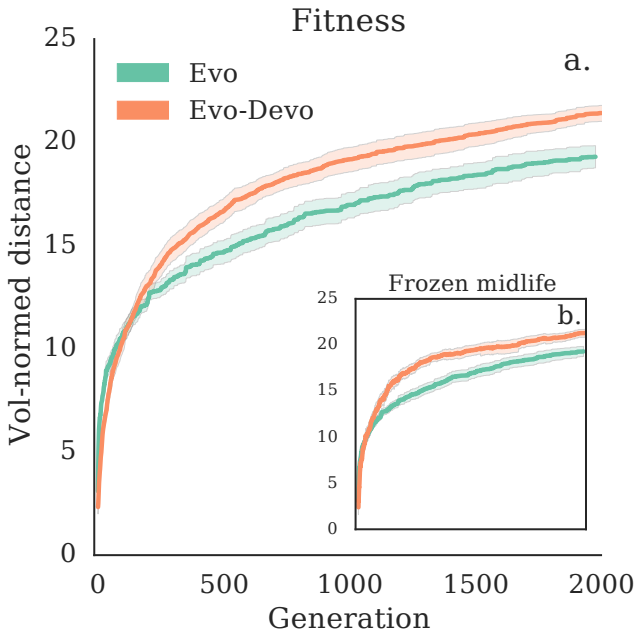


Figure 3.4: For thirty runs, a population of thirty robots is evolved for two thousand generations. (a) Best of generation fitness for Evo and Evo-Devo robots. (b) The same robots are reevaluated with development frozen at their midlife morphology. Means are plotted with 95% bootstrapped confidence intervals.

displacement may be lower in the Evo-Devo case, since these robots 'travel' through a number of different morphologies, they may pass through those which run at a higher instantaneous speed (but spend less of their lifetime in this morphology).

Evolution.

The results of the evolutionary algorithm are displayed in figure 3.4a. In the earliest generations, evolution is consistent with random search and the best Evo robots start off slightly better than the best Evo-Devo robots. However, the best Evo-Devo robots quickly overtake the best Evo robots. At the end of optimization there is a significant difference between Evo and Evo-Devo run champions ($U = 122,\ p < 0.001$).

We also chose to reevaluate Evo-Devo robots with their development frozen at their median ontogenetic morphologies (figure 3.4b). For each robot, we measure the robot's fitness (equation 3.10) at this midlife morphology with development frozen, for two seconds. Selection is completely blind to this frozen evaluation. It exists solely for the purpose of post-evolution analysis, and serves primarily as a sanity check to make sure Evo-Devo robots are not explicitly utilizing their ability to grow/shrink to

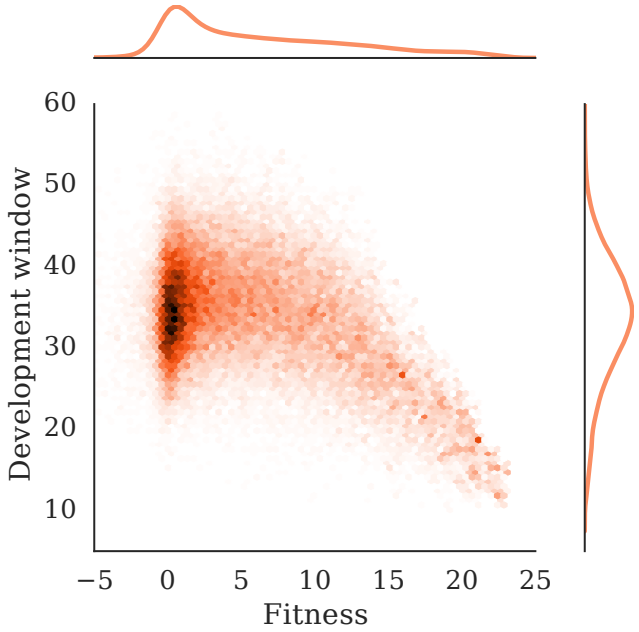


Figure 3.5: The relationship between the amount of development at the individual level (W) and fitness (F). The fastest individuals have small developmental windows surrounding a fast body plan.

move faster.

Development appears to inhibit locomotion to some degree as the best morphologies run slightly faster with development turned off, particularly in earlier generations. A significant difference, at the 0.001 level, between Evo robots and Evo-Devo robots with development frozen at midlife, occurs after only 108 generations compared to 255 generations with development enabled. Note that the midlife morphology is not necessarily the top speed of an Evo-Devo robot. In fact it is almost certainly not the optimal ontogenetic form since the best body plan may occur at any point in its continuous ontogeny, including the start and endpoints.

Closing the window.

Once an Evo-Devo robot identifies a good body plan in its ontogenetic sweep, its descendants can gain fitness by 'suppressing' development around the good plan through heterochronic mutations. This can be accomplished by incrementally closing the *developmental window*, the interval $(s_{k0},\ s_{k1})$, for each voxel, around the good morphology. In the limit, under a fixed environment, this process ends with a decedent born with the good design from the start and devoid of any developmental at all ($s_{k0} = s_{k1}$

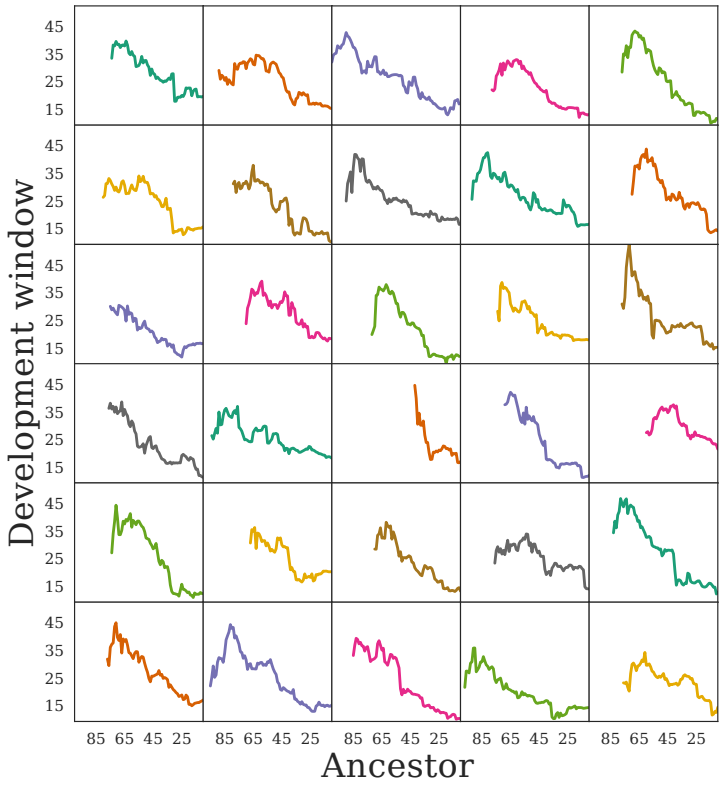


Figure 3.6: Closing the window. Total development window trajectories (in phylogeny) of the lineages of the most fit individuals in each run. Phylogenetic time goes from left to right: from the oldest ancestor (randomly created) to its most recent decedent, the current run champion.

for all voxels). This phenomenon, best known as the Baldwin Effect, is instrumental in evolution because natural selection is a hill-climbing process and therefore blind to needles in a haystack, good designs (local optima) to which no gradient of increased fitness leads. The developmental sweep, however, alters the search space in which evolution operates, surrounding the good design by a slope which natural selection can climb [96].

To investigate the relationship between development and fitness, we add up all of the voxel-level development windows to form a individual-level summary statistic, W. We define the *total* development window, W, as the sum of the absolute difference of starting and final resting lengths across the robot's 48 voxels.

$$W = \sum_{k=1}^{48} \text{abs}(s_{k1} - s_{k0}) \tag{3.11}$$

Overall there is a strong negative correlation between fitness, F, and the total

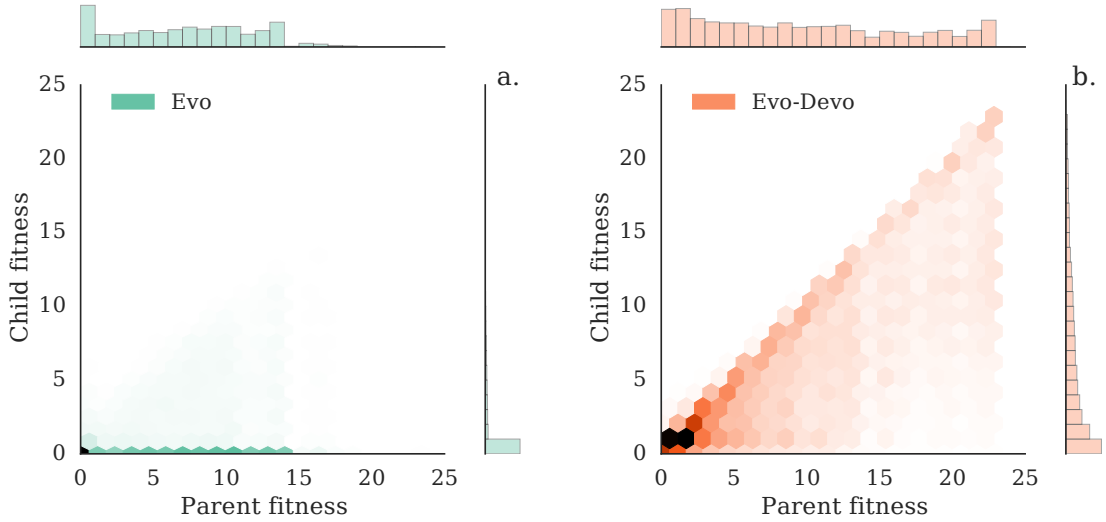


Figure 3.7: Mutation impact: child fitness by parent fitness (vol-normed distance). The diagonal represents a neutral mutation, equivalent child and parent fitness. Hexagon bins below the diagonal represent detrimental mutations (child less fit than its parent); bins above the diagonal represent beneficial mutations (child more fit than its parent).

development window, W, in Evo-Devo robots (figure 3.5). To achieve the highest fitness values a robot needs to have narrow developmental windows at the voxel level. However, this statistic doesn't discriminate between open/closed windows early/late in evolution. To show what sorts of development window/evolutionary time relationships eventually lead to highly fit individuals, we grab the lineages of only the most fit individuals at the end of evolutionary time (figure 3.6). In the most fit individuals, development windows tend to first increase slightly in phylogeny before decreasing to their minimum, or close nearby. The age objective in AFPO lowers the selection pressure on younger individuals which allows them to explore, through larger developmental windows, a larger portion of design space until someone in the population discovers a locally optimal solution which creates a new selection pressure for descendants with older genetic material to 'lock in' or canalize this form with smaller developmental windows. These results further suggests that development itself is not optimal, it is only helpful in that it can lead to better optima down the road once the window is closed.

The effect of mutations.

In addition to the parameter-sweeping nature of its search, developmental time provides evolution with a simple mechanism for inducing mutations with a range of

magnitude of phenotypic impact. The overall mutation impact in our experiments is conveyed in figure 3.7 through 2D histograms of child and parent fitness. Recall that a child is created through mutation by each individual (parent) in the current population. These plots include the entire evolutionary history of all robots in every run. There are relatively so few robots with negative fitness that the histograms need not extend into this region since they contain practically zero density and would appear completely white.

The diagonal represents equal parent and child fitness, a behaviorally neutral mutation. Hexagons below the diagonal represent detrimental mutations: lower child fitness relative to that of its parent. Hexagons above the diagonal represent beneficial mutations: higher child fitness relative to that of its parent. Mutations are generally detrimental for both groups, particularly in later generations once evolution has found a working solution. For Evo robots (figure 3.7a), most if not all of the mass in the marginal density of child speed is concentrated around zero. This means that mutations to an Evo robot are almost certain to break the existing parent solution, rendering a previously mobile design immobile.

The majority of Evo-Devo children, however, are generally concentrated on, or just below the diagonal in figure 3.7b. This general pattern holds even in later generations when evolution has found working solutions with high fitness. It follows that mutations to an Evo-Devo robot may be phenotypically smaller than mutations to an Evo robot, even though they use the same mutation operator. Furthermore, figure 3.7b displays a high frequency of mutations with a wide range of magnitude of phenotypic impact including smaller, low-risk mutations which are useful for refining mobile designs; as well as a range of larger, higher-risk mutations which occasionally provide the high-reward of jumping into the neighborhood of a more fit local optima at a range of distances in the fitness landscape.

Now let's define the impact of developmental mutations, M, as the relative difference in child (F_C) and parent fitnesses (F_P), for positive fitnesses only.

$$M = \frac{F_C}{F_P} - 1; \qquad F_C,\ F_P > 0 \tag{3.12}$$

Then the average mutational impact for early-in-the-life mutations (any mutations that, at least in part, modify initial volumes) is $M_0 = -0.29$. While the average mutational impact for late-in-the-life mutations (that modify final volumes) is $M_1 = -0.10$. Although both types of mutations are detrimental on average, later-in-life mutations are more beneficial (less detrimental) on average ($p < 0.001$). This makes sense in a task with dependent time steps since a child created through a late-in-life mutation will at least start out with the same behavior as its parent and then slowly diverge over its life. Whereas an early-in-life mutation creates a behavioral change at $t = 0$.

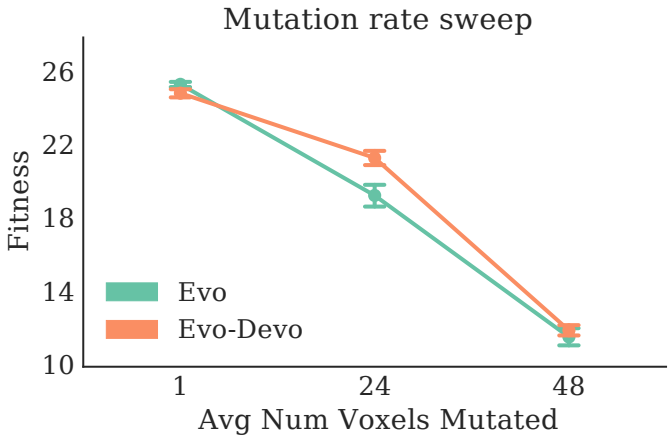


Figure 3.8: A hyperparameter sweep of mutation rate: a probability dictating the average number of voxels mutated in an individual robot.

The necessity of development.

In attempting to induce a needle-in-the-haystack fitness landscape, as a proof of concept, we intentionally set the mutation rate and scale fairly high. A low-resolution hyperparameter sweep (figure 3.8) indicates that the efficacy of ballistic development is indeed dependent on the mutation rate: there is no significant difference between Evo and Evo-Devo at either very low or very high rates. Higher fitness values are obtained through smaller mutation rates, which raises the question: Is development useful only in its ability to decrease the phenotypic impact of mutations? If so we might prefer Evo robots (with a low mutation rate) since they reside in a smaller search space. But how low should the mutation rate be? It may in fact be difficult to know *a priori* which mutation rate is optimal. It is also important to recognize that while we use mutation rate here to artificially tune the ruggedness of the fitness landscape, in a naturally rugged landscape we presumably would not have direct access to such an easily tunable parameter to 'undo', or smooth-out the ruggedness.

Moreover, we know that there exist contexts in which developmental flexibility can permit the local speeding up of the basic, slow process of natural selection, thanks to the Baldwin Effect [62]. Our new data suggests that even open-loop morphological change increases the probability of randomly finding (and subsequently 'locking in') a mobile design (figure 3.3), and that this probability is increasing in the amount of change (figure 3.6) even though ballistic development and fitness are inversely correlated (figure 3.5). The staticity of Evo robots prevents this local speed-up which can place them at a significant disadvantage in rugged fitness landscapes.

3.4 Conclusion

In this paper we introduced a minimal yet embodied model of development in order to isolate the intrinsic effect of morphological change in ontogenetic time, without the confounding effects of environmental mediation. Even our simple developmental model naturally provides a continuum in terms of the magnitude of mutational phenotypic impact, from the very large (caused by early-in-life developmental mutations) to the very small (caused by late-in-life mutations). We predict that, because of this, such a developmental system will be more evolvable than an equivalent non-developmental system because the latter lacks this inherent spectrum in the magnitude of mutational impacts.

We showed that even without any sensory feedback, open-loop development can confer evolvability because it allows evolution to sweep over a much larger range of body plans. Our results suggest that widening the span of the developmental sweep increases the likelihood of stumbling across locally optimal designs otherwise invisible to natural selection, which automatically creates a new selection pressure to canalize development around this good form. This implies that species with completely blind developmental plasticity tend to evolve faster and more 'clearsightedly' than those without it.

Future work will involve closing the developmental feedback loop with as little additional machinery as possible to determine when and how such added complexity increases evolvability.

Chapter 4

Shape and Configuration

Appeared as:
S. Kriegman et al., How morphological development can guide evolution. *Scientific Reports* **8** (1), 1–10 (2018).

Abstract:
Organisms result from adaptive processes interacting across different time scales. One such interaction is that between development and evolution. Models have shown that development sweeps over several traits in a single agent, sometimes exposing promising static traits. Subsequent evolution can then canalize these rare traits. Thus, development can, under the right conditions, increase evolvability. Here, we report on a previously unknown phenomenon when embodied agents are allowed to develop and evolve: Evolution discovers body plans robust to control changes, these body plans become genetically assimilated, yet controllers for these agents are not assimilated. This allows evolution to continue climbing fitness gradients by tinkering with the developmental programs for controllers within these permissive body plans. This exposes a previously unknown detail about the Baldwin effect: instead of all useful traits becoming genetically assimilated, only traits that render the agent robust to changes in other traits become assimilated. We refer to this as *differential canalization*. This finding also has implications for the evolutionary design of artificial and embodied agents such as robots: robots robust to internal changes in their controllers may also be robust to external changes in their environment, such as transferal from simulation to reality or deployment in novel environments.

4.1 Introduction

The shape of life changes on many different time scales. From generation to generation, populations gradually increase in complexity and relative competency. At the individual level, organisms grow from a single-celled egg and exhibit extreme postnatal change as they interact with the outside world during their lifetimes. At a faster

time scale still, organisms behave such as to survive and reproduce.

Many organisms manifest different traits as they interact with their environment. It seems wasteful not to utilize this extra exploration to speed the evolutionary search for good genotypes. However, to communicate information from these useful but temporary traits to the genotype requires inverting the generally very complex, nonlinear and stochastic mapping from DNA to phenotype. Inverting such a function would be exceedingly difficult to compute. Organisms can, however, pass on their particular capacity to acquire certain characteristics. Thus phenotypic plasticity can affect the direction and rate of evolutionary change by influencing selection pressures. Although this phenomenon was originally described by Baldwin [12], Morgan [161] and Waddington [244], among others, it has become known as 'the Baldwin effect'. In Baldwin's words: 'the most plastic individuals will be preserved to do the advantageous things for which their variations show them to be the most fit, and the next generation will show an emphasis of just this direction in its variations' [12]. In a fixed environment, when the 'advantageous thing' to do is to stay the same, selection can favor genetic variations which more easily, reliably, or quickly produce these traits. This can lead to the genetic determination of a character which in previous generations needed to be developed or learned.

Thirty years ago, Hinton and Nowlan [96] provided a simple computational model of the Baldwin effect that clearly demonstrated how phenotypic plasticity could, under certain conditions, speed evolutionary search without communication to the genotype. They considered the evolution of a bitstring that is only of value when perfectly matching a predefined target string. The search space therefore has a single spike of high fitness with no slope leading to the summit. In such a space, evolution is no better than random search.

Hinton and Nowlan then allowed part of the string to randomly change at an additional (and faster) developmental time scale. When the genetically specified (nonplastic) portion of the string is correct, there is a chance of discovering the remaining portion in development. The speed at which such individuals tend to find the good string will be proportional to the number of genetically determined bits. When the target string is found, development stops and the individual is rewarded for the amount of remaining developmental time. This has the effect of creating a gradient of increasing fitness surrounding the correct specification that natural selection can easily climb by incrementally assimilating more correct bits to the genotype.

Hinton and Nowlan imagined the bitstring as specifying the connections of a neural network in a very harsh environment. We are also interested in this interaction of subsystems unfolding at different time scales, but consider an embodied agent situated in a physically-realistic environment rather than an abstract control system. This distinction is important as it grounds our hypotheses in the constraints and opportunities afforded by the physical world. It also allows us to investigate how changes

in morphology and control might differentially affect the direction or rate of evolutionary search. More specifically, it exposes the previously unknown phenomenon of differential canalization reported here.

Inspired by Hinton and Nowlan, Floreano and Mondada [77] explored the interaction between learning and evolution in mobile robots with a fixed body plan but plastic neural control structure. They noted that the acquisition of stable behavior in ontogeny did not correspond to stability (no further change) of individual synapses, but rather was regulated by continuously changing synapses which were dynamically stable. In other words, agents exploited this ontogenetic change for behavior, and this prevented its canalization. In this paper, we structure development in a way that restricts its exploitation for behavior and thus promotes the canalization of high performing static phenotypes. Also, the robot's body plan was fixed in Floreano and Mondada's experiments[77], whereas in the work reported here, evolution and development may modify body plans.

Several models that specifically address morphological development of embodied agents have been reported in the literature [29, 60, 64, 68, 154]. However, the relationship between morphological development and evolvability is seldom investigated in such models. Moreover, there are exceedingly few cases that considered postnatal change to the body plan of an agent (its resting structural form) as it behaves and interacts with the environment through physiological functioning (at a faster time scale).

We are only aware of four cases reported in the literature in which a simulated robot's body was allowed to change while it was behaving. In the first two cases [114, 243], it was not clear whether this ontogenetic morphological change facilitated the evolution of behavior. Later, Bongard [28] demonstrated how such change could lead to a form of self-scaffolding that smoothed the fitness landscape and thus increased evolvability. This ontogenetic change also exposed evolution to a wider range of sensor-motor contingencies, which increased robustness to novel environments. More recently, Kriegman *et al.* [121] showed how development can sweep over a series of body plans in a single agent, and subsequent heterochronic mutations canalize the most promising body plan in more morphologically-static descendants.

We are not aware of any cases reported in the literature to date in which a simulated robot's body and control are simultaneously allowed to change while it is behaving. In this paper, we investigate such change in the morphologies and controllers of soft robots as they are evolved for coordinated action in a simulated 3D environment. By morphology we mean the current state of a robot's shape, which is slowly changed over the course of its lifetime by a developmental process. We distinguish this from the controller, which sends propagating waves of actuation throughout the individual, which also affects the instantaneous shape of the robot but to a much smaller degree. We here refer to these two processes as 'morphology' and 'control'.

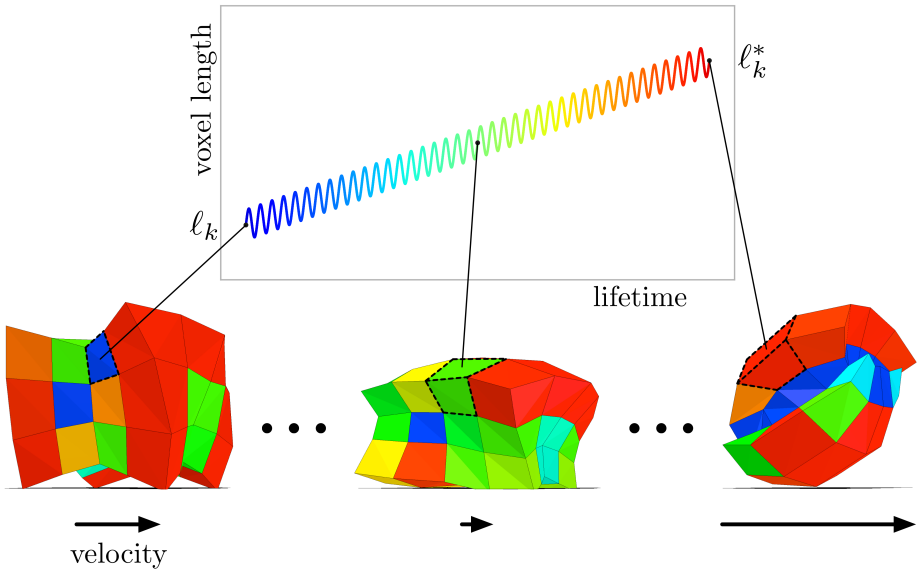


Figure 4.1: **Modeling development.** An evolved soft robot changes its shape during its lifetime (post-natal development), from a walking quadruped into a rolling form. Evolution dictates how a robot's morphology develops by setting each voxel's initial (ℓ_k) and final (ℓ_k^*) resting length. The length of a single voxel k is plotted to illustrate its (slower) growth and (faster) actuation processes. Voxel color indicates the current length of that cell: the smallest voxels are blue, medium sized voxels are green, and the largest voxels are red. As robots develop and interact with a physically realistic environment, they generate heterogeneous behavior in terms of instantaneous velocity (bottom arrows). Soft robot evolution, development and physiological functioning can be seen in Supplementary Video S1.

As both processes change the shape, and thus behavior, of the robot, this distinction is somewhat arbitrary. However, the central claim of this paper, which is that some traits become canalized while others do not, is not reliant on this distinction.

We use soft robots because they provide many more degrees of morphological freedom compared to traditional robots composed of rigid links connected by rotary or linear actuators. This flexibility allows soft robots to accomplish tasks that would be otherwise impossible for their rigid-bodied counterparts, such as squeezing through small apertures [44] or continuously morphing to meet different tasks. Recent advancements in materials science are enabling the fabrication of 3D-printed muscles [155] and nervous systems [247]. However, there are several challenges to the field of soft robotics, including an overall lack of design intuition: What should a robot with nearly unbounded morphological possibility look like, and how can it be controlled? Controllability often depends on precision actuation and feedback authority, but these properties are difficult to maintain in soft materials in which motion in one part of the robot can propagate in unanticipated ways throughout its body [139].

We present here a minimally complex but embodied model of morphological and

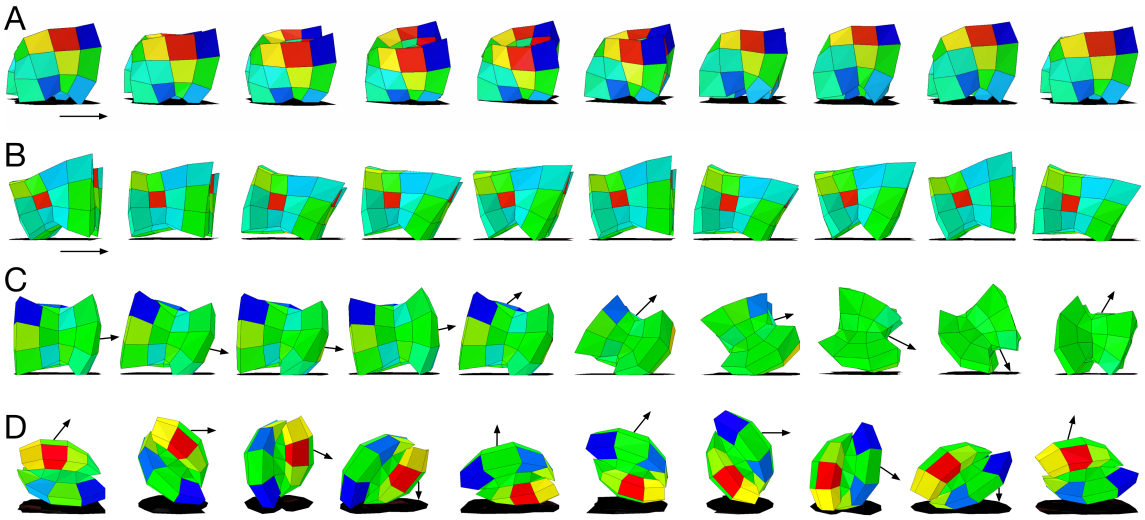


Figure 4.2: **Evolved behavior.** Each row depicts a different evolved robot moving from left to right. Voxels in this figure are colored by the amount of subsequent morphological development remaining at that cell: blue indicates shrinking voxels ($\ell_k > \ell_k^*$), red indicates growing voxels ($\ell_k < \ell_k^*$), green indicates little to no change either way ($\ell_k \cong \ell_k^*$). (A) An evolved trotting soft quadruped with a two-beat gait synchronizing diagonal pairs of legs. (B) A galloping adult robot which goes fully airborne mid-gait. (C) A galloping juvenile robot which develops into a rolling adult form. (D) A rolling juvenile robot at 10 points in ontogeny immediately after birth. Arrows indicate the general directionality of movement, but this is more precisely captured by Supplementary Video S1.

neurological development. This new model represents an alternative approach to the challenging problem of soft robot design and presents an *in silico* testbed for hypotheses about evolving and developing embodied systems. This model led to the discovery of differential canalization and how it can increase evolvability.

4.2 Results

We consider a locomotion task, over flat terrain, for soft robots composed of a $4 \times 4 \times 3$ grid of voxels (Fig. 4.1). Robots are evaluated for 40 actuation cycles at 4 Hz, yielding a lifetime of ten seconds. Fitness is taken to be distance traveled measured in undeformed body lengths (four unit voxels, i.e. 4 cm). Example robots are shown in Figures 4.1 and 4.2, and Supplementary Video S1.

All experiments were performed in the open-source soft-body physics simulator *Voxelyze* [95]. Voxelyze simulates soft materials using two elements: particles and beams. A particle is a point mass with rotational inertia. A spring-like beam (with translational and rotational stiffness) connects two adjacent particles. Each particle is connected to at most six neighbors (above, below, front, back, left, and right), on a

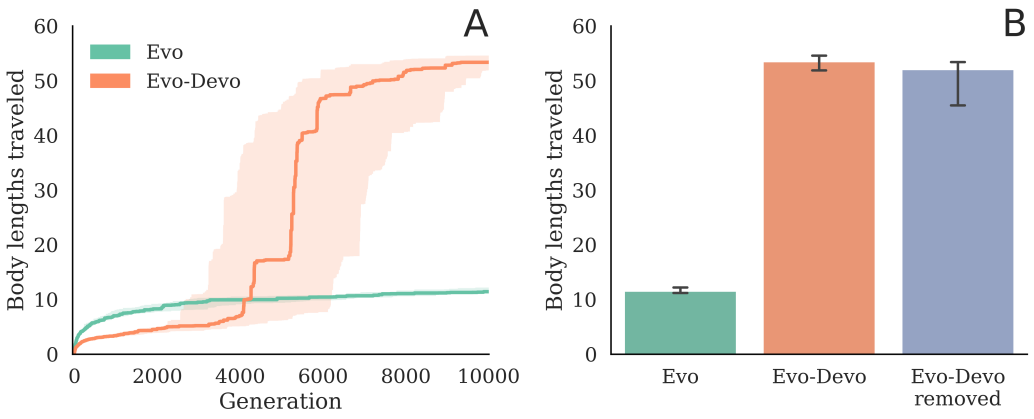


Figure 4.3: **Evolvability and development.** Morphological development drastically increases evolvability (A), even when development is manually removed from the evolved systems (the run champions) by setting the final parameter values equal to their starting values ($\ell_k^* = \ell_k$ and $\phi_k^* = \phi_k$), in each voxel (B). Median fitness is plotted with 95% bootstrapped confidence intervals for three treatments: evolving but non-developmental robots (Evo), evolving and developing robots (Evo-Devo), and evolving and developing robots evaluated at the end of evolution with their development removed (Evo-Devo removed). Fitness of just the final, evolved populations (at generation 10000) are plotted in B.

cartesian grid. Local material properties are stored at particles and averaged across shared beams. Finally, a voxel mesh is drawn around each particle for visualization, such that adjacent voxels touch at the center of their shared beam. More details about how this is actually implemented are given by Hiller and Lipson [95].

The morphology of a robot is given by the resting (beam) length stored at each voxel (Fig. 4.1). However the shape and volume of each voxel is changed by external forces from the environment and internal forces via behavior. The morphology of a robot is denoted by the $4 \times 4 \times 3 = 48$-element vector ℓ, where each element is the resting length stored at that voxel (with possible values within 1.0 ± 0.75 cm). Like most animals, our robots are bilaterally symmetrical. We built this constraint into our robots because bilateral symmetry is known to help with forward locomotion [87]. The lefthand $2 \times 4 \times 3 = 24$ resting voxel lengths are reflected on the other, righthand side of the midsagittal line, yielding 24 independent resting lengths.

The controller, however, is not constrained to be symmetrical since many behaviors, even for symmetric morphologies, consist of asymmetric gaits, and is given by the phase offset of each voxel from a global oscillating signal with an amplitude of 0.14 cm. The controller is denoted by the 48-element vector ϕ, where each element is the phase offset of that voxel (with possible values within $0 \pm \pi/2$).

We investigated the impact of development in this model by comparing two experimental variants: Evo and Evo-Devo (schematized in Supplementary Fig. 4.8). The control treatment, **Evo**, lacks development and therefore maintains a fixed morphol-

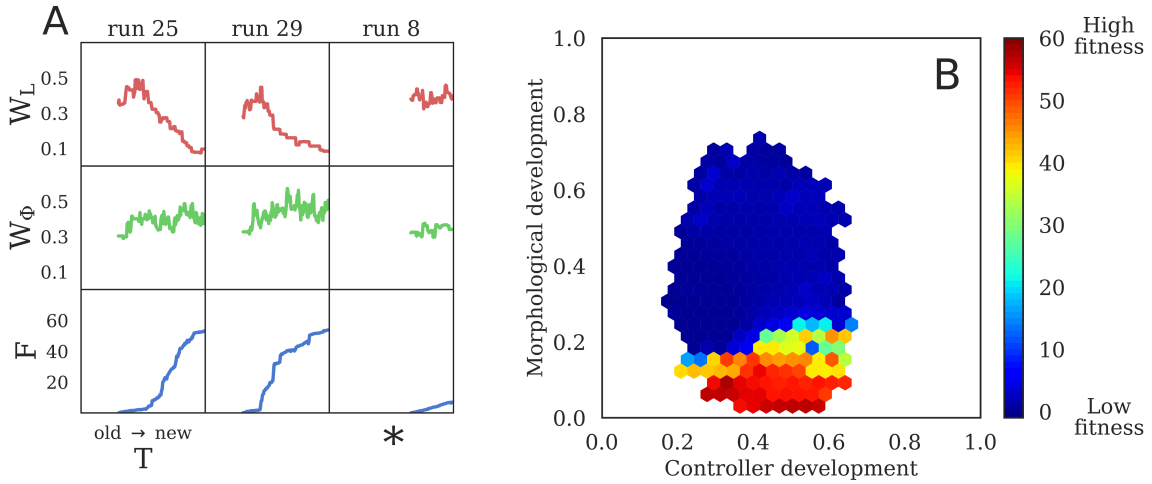


Figure 4.4: **Differential canalization.** Developmental windows (i.e. the total lifetime developmental change) for morphology, W_L (see Equation 4.5), and controller, W_Φ (see Equation 4.6), alongside fitness F. (A) Three representative lineages taken from Supplementary Fig. 4.7, which displays the lineages of all 30 Evo-Devo run champions. Evolutionary time T moves from the oldest ancestor (left) to the run champion (right). A general trend emerges wherein lineages initially increase their morphological development in T (rising red curves) and subsequently decrease morphological development to almost zero (falling red curves). Five of the 30 evolutionary trials, annotated by $*$, fell into a local optima. (B) Median fitness as a function of morphology and controller development windows (W_L, W_Φ), for all Evo-Devo designs evaluated. Overall, the fastest designs tend to have small amounts of morphological development, but are free to explore alternative control policies.

ogy and control policy in a robot as it behaves over its lifetime. Two parameters per voxel are sufficient to specify an evolved robot at any time t in its lifetime: its morphology ℓ_k, and controller ϕ_k. An evolutionary algorithm optimizes 24 morphological and 48 control parameters.

The experimental treatment **Evo-Devo** evolves a developmental program rather than a static phenotype (Fig. 4.1). For each parameter in an Evo robot, an Evo-Devo robot has two: its starting and final value. The evolutionary algorithm associated with the Evo-Devo treatment thus optimizes 48 morphological and 96 control parameters. The morphology and controller of the k-th voxel change linearly from starting to final values, throughout the lifetime of a developing robot. The endpoint parameters are denoted by asterisks: the controller develops from ϕ to ϕ^*, the morphology develops from ℓ to ℓ^*. The starting and final points of development are predetermined by a genome which in turn fixes the direction (compression or expansion) and rate of change for each voxel. Development is thus *ballistic* in nature rather than adaptive, as it cannot be influenced by the environment (Equation 4.1).

For both treatments we conducted 30 independent evolutionary trials. At the end of evolutionary optimization, the non-developmental robots (Evo) tend to move on

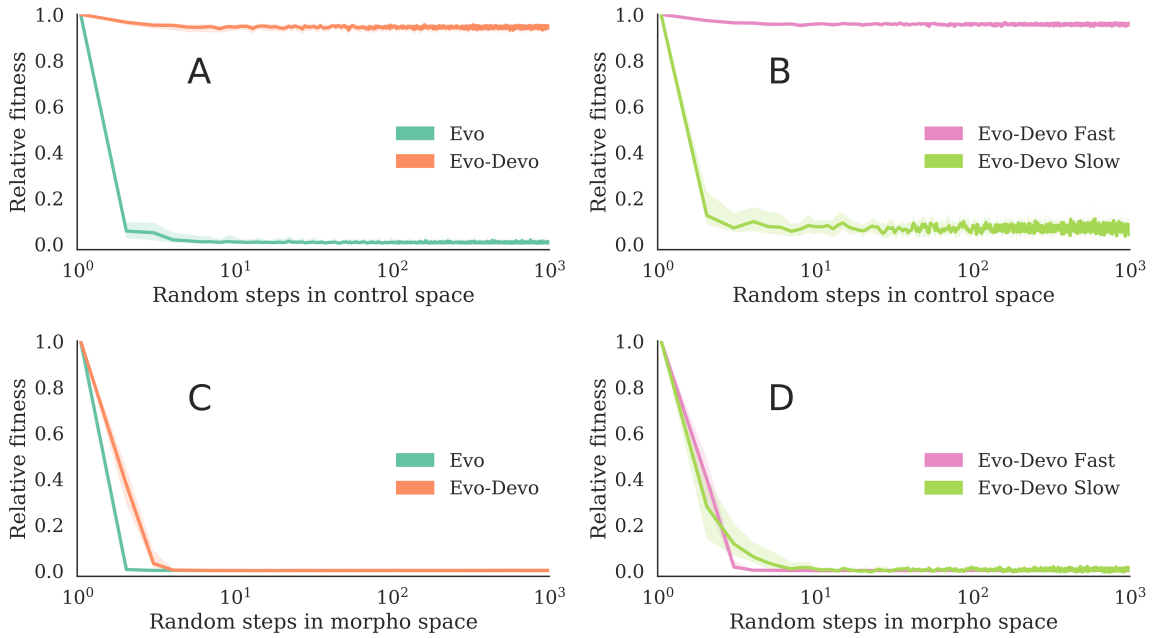


Figure 4.5: **Sensitivity to morphological and control mutations.** Ten random walks were taken from each run champion. (A) Successive *control* mutations to the Evo and Evo-Devo run champions. (B) The previous Evo-Devo results separately for fast and slow design types. (C) Successive *morphological* mutations to the Evo and Evo-Devo run champions. (D) The previous Evo-Devo results separately for fast and slow design types. Medians plotted with 99% confidence intervals. The faster Evo-Devo robots tend to possess body plans that are robust to control mutations.

average with a speed of 10 body lengths in 10 seconds, or 1 length/sec. The evolved and developing robots (Evo-Devo) tend to move at over 5 lengths/sec (Fig. 4.3A). To ensure evolved and developing robots are not exploiting some unfair advantage conferred by changing body plans and control policies unavailable to non-developmental robots, we manually remove their development by setting $\ell^* = \ell$ and $\phi^* = \phi$, which fixes the structure of their morphologies and controllers at birth ($t = 0$) (Equation 4.1). That is to say, we convert the evolved Evo-Devo robots into Evo robots (Equation 4.1 reduces to Equation 4.2). The resulting reduced robots suffer only a slight (and statistically non-significant) decrease in median speed and still tend to be almost five times faster than the systems evolved without development (Fig. 4.3B, treatment 'Evo-Devo removed'). Ballistic development is therefore beneficial for search but does not provide a behavioral advantage in this task environment.

To investigate this apparent search advantage, we trace development and fitness across the 30 lineages which produced a 'run champion': the robot with highest fitness at the termination of a given evolutionary trial (Supplementary Fig. 4.7).We measure the amount of ballistic change in each robot—its 'ballistic plasticity'—by a statistic

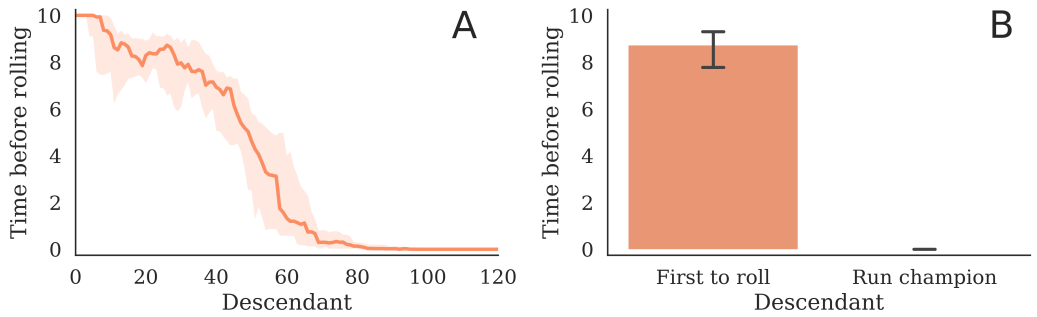


Figure 4.6: **Late onset discoveries.** Ontogenetic time before the discovery of rolling over, taken from the lineages of the best robot from each of the 25 Evo-Devo trials that produced a rolling design. Median time to discovery, with 95% C.I.s, for (A) the lineage from the most distant ancestor ($T = 0$) to more recent descendants, and (B) the first ancestor to roll over compared to the final run champion. Rolling over is measured from the first time step the top of the robot touches the ground, rather than after completely rolling over. The first ancestors to roll over tend to do so at the end of their lives, their descendants tend to roll sooner in life, and the final run champions all begin rolling immediately at birth. These results are a consequence of dependent time steps: because mutational changes affect all downstream steps, their phenotypic impact is amplified in all but the terminal stages of development. Thus, late onset changes can provide exploration in the search space without breaking rest-of-life functionality, and subsequent evolution can gradually assimilate this trait to the start of development.

we call the *developmental window*. The developmental window is defined separately for morphology (Equation 4.5) and control (Equation 4.6) as the absolute difference in starting and final values summed across the robot and divided by the total amount of possible development, such that 0 and 1 indicate no and maximal developmental change, respectively. Evo robots by definition have development windows of zero, as do Evo-Devo robots that have had development manually removed. An Evo-Devo robot with a small developmental window has thus become *canalized* [244].

In terms of fitness, there were two observed basins of attraction in average velocity: a slower design type which either trots or gallops at a speed of less than 1 length/sec (Fig. 4.2A,B and Fig. 4.4A*), and a faster design type that rolls at 5-6 lengths/sec (Fig. 4.2C). After ten thousand generations, 25 out of a total of 30 Evo-Devo trials (83.3%) find the faster design, compared to just 6 out of 30 Evo trials (20%).

Differential canalization.

Modular systems are more evolvable than non-modular systems because they allow evolution to improve one subsystem without disrupting others [138, 245]. Modularity may be a property of the way a system is built, or it may be an evolved property. The robots evolved here are by definition modular because the genes which affect morphology are independent of those which affect its control. However the more

successful Evo-Devo lineages evolved an additional form of modularity, which we term differential canalization: Some initially developmentally plastic traits become integrated and canalized, while other traits remain plastic.

In the successful Evo-Devo trials, morphological traits were canalized while control traits were not. Evidence for this is provided in Supplementary Fig. 4.7,which is summarized by Fig. 4.4A. Trajectories of controller development (green curves) do not follow any discernible pattern in phylogenetic time, and appear upon visual inspection to be consistent with a random walk or genetic drift. The trajectories of morphological development (red curves), however, follow a consistent pattern. The magnitude of morphological development increases slightly, but significantly ($p < 0.001$), before decreasing all the way to the most recent descendant, which is the most fit robot from that trial (the run champion). Run champions tend to have much less morphological development than their most distant ancestor ($p < 0.001$), but there is not a significant difference between champion and ancestral controller windows. Furthermore, this pattern tends to correlate with high fitness: in trials in which this pattern did not appear (runs 6, 8, 16-18), fitness did not increase appreciably over evolutionary time.

This process within the lineages of the run champions is consistent with a more general correlation found in all designs explored during optimization across all runs: Individuals with the highest fitness values tend to have very small amounts of morphological development, while their control policies are free to develop (Fig. 4.4B). However, despite the fact that morphological development tends to be canalized in the most fit individuals, it cannot simply be discarded as the non-developmental systems have by definition small morphological windows, and small controller windows, but also low fitness.

To test the sensitivity of the evolved morphologies to changes in their control policies, we applied a random series of control mutations to the Evo and Evo-Devo run champions from each evolutionary trial. For each run champion, we perform 1000 subsequent random controller mutations that build upon each other in series (a Brownian trajectory in the space of controllers)—and repeat this process ten times for each run champion, each with a unique random seed. It was found that optimized Evo-Devo robots tend to possess body plans that are much more robust to control mutations than those of Evo robots (Fig. 4.5A). The first control mutation to optimized Evo robots tends to immediately render them immobile, whereas optimized Evo-Devo robots tend to retain most of their functionality even after 1000 successive random changes to their controllers. Within Evo-Devo designs, the functionality of the 25 fast designs are minimally affected by changes to their control, whereas the five slow designs also tend to break after the first control mutation (Fig. 4.5B). Thus it can be concluded that these five robots are non-modular: their non-canalized morphologies evolved a strong dependency on their controllers. The Evo robots are

similarly non-modular: they are brittle to control mutations.

To test the sensitivity of the evolved controllers to changes in their morphologies, we applied the same procedure described in the previous paragraph but with random morphological mutations rather than control mutations. It was found that both developmental and non-developmental systems tend to evolve controllers that are very sensitive to morphological mutations (Fig. 4.5C). These findings are consistent with those of Cheney *et al.* [46], who also reported that robots were more sensitive to changes in their morphology than in their controllers. Here, the first few morphological mutations to optimized robots, in both treatments, tend to immediately render them immobile. Within Evo-Devo design types, neither of which canalized development in their controllers (Supplementary Fig. 4.7,both the fast and slow designs possess controllers sensitive to changes in their morphologies (Fig. 4.5D). Thus it can be concluded that the non-canalized controllers evolved a strong dependency on their morphologies. Therefore the only trait to be successfully canalized was also the only trait that rendered the agent robust to changes in other traits.

Heterochrony in morphological development.

The evolutionary algorithm can rapidly discover an actuation pattern that elicits a very small amount of forward movement in these soft robots regardless of the morphology. There is then an incremental path of increasing locomotion speed that natural selection can climb by gradually growing legs to reduce the surface area touching the floor and thus friction, and simultaneously refining controller actuation patterns to better match and exploit the morphology (Fig. 4.2A,B).

There is, however, a vastly superior design partially hidden from natural selection—a 'needle in the haystack', to use Hinton and Nowlan's metaphor [96]. On flat terrain, rolling can be much faster and more efficient than walking, but finding such a design is difficult because the fitness landscape is deceptive. Rolling over once is much less likely to occur in a random individual than shuffling forwards slightly. And as a population continues to refine walking morphologies and gaits, lineages containing rocking individuals which are close to rolling over, or roll over just once, do not survive long enough to eventually produce a true rolling descendant.

Development can alter the search space evolution operates in because individuals sweep over a continuum of phenotypes, with different velocities, rather than single static phenotype that travels at a constant speed (Supplementary Fig. 4.8E,J).The lineages which ultimately evolved the faster rolling design initially increased their morphological plasticity in phylogenetic time as evidenced by the initial upward trends in the red curves in Supplementary Fig. 4.7(summarized by Fig. 4.4A) which contain a statistically significant difference between their starting and maximum developmental window sizes ($p < 0.001$). This exposes evolution to a wider range of body plans and

thus increases the chance of randomly rolling at least once at some point during the evaluation period.

The peak of morphological plasticity in Supplementary Fig. 4.7(summarized by Fig. 4.4A) generally lines up with the start of an increasing trend in fitness (blue curves) and marks the onset of differential canalization. Rolling just once allows an individual to move further (1 body length) than some early walking behaviors but they incur the fitness penalty of having fallen over and thus not being able to subsequently walk for the rest of the trial. Therefore this tends to happen at the very end of ontogeny (Fig. 4.6), as individuals evolve to 'dive' in the last few time steps of the simulation of their behavior, thus incurring an additional increase of fitness over their parent, which does not exhibit this behavior. Since more rolling incurs more fitness than less rolling, a form of progenesis occurs as heterochronic mutations move ℓ_k closer to ℓ_k^*, for each voxel. This gradually earlifies rolling from a late onset behavior to one that arises increasing earlier in ontogeny (Supplementary Video S1). As more individuals in the population discover and earlify this rolling behavior, the competition stiffens until eventually individuals which are not born rolling from the start are not fast enough to compete (Fig. 4.2C,D).

Generality of results.

For the results above, as in nature [143], the mutation rate of each voxel was left under evolutionary control (self-adaptation). In an effort to assess the generality of our results, we replicated the experiment described above for various fixed mutation rates (Supplementary Fig. 4.9).Without development, as in Hinton and Nowlan's case[96], the search space has a single spike of high fitness. One can not do better than random search in such a space. At the highest mutation rate, optimizing Evo morphologies reduces to random search, and this is the only mutation rate where Evo does not require significantly more generations than Evo-Devo to find the faster design. This can be observed in Supplementary Fig. 4.9 by comparing the generation at which the slopes of the fitness curves increase dramatically. However, the best two treatments, as measured by the highest median speed at the end of optimization, have development, and the robots they produced are significantly faster than those produced by random search (Evo with the highest mutation rate) ($p < 0.01$).

To further test the sensitivity of our results to the various settings of our particular system, we transcribed the main experiment for a different class of morphologies (rigid bodies) and controllers (neural networks). Details are provided in Supplementary Fig. 4.11 and Supplementary Methods. The results of this test indicate that differential canalization exists elsewhere, but it does not always increase evolvability.

4.3 Discussion

In these experiments, the intersection of two time scales—slow linear development and rapid oscillatory actuation, as from a central pattern generator—generates positive and negative feedback in terms of instantaneous velocity: the robot speeds up and slows down during various points in its lifetime (Supplementary Fig. 4.8J).Prior to canalization, unless all of the phenotypes swept over by an individual in development keep the robot motionless, there will be intervals of relatively superior and inferior performance. Evolution can thus improve overall fitness in a descendant by lengthening the time intervals containing superior phenotypes and reducing the intervals of inferior phenotypes. However, this is only possible if such mutations exist.

We have found here that such mutations do exist in cases where evolutionary changes to one trait do not disrupt the successful behavior contributed by other traits. For example, robots that exhibited the locally optimal trotting behavior (Fig. 4.2A) exhibited a tight coupling between morphology and control, and thus evolution was unable to canalize development in either one, since mutations to one subsystem tended to disrupt the other. Brief ontogenetic periods of rolling behavior (Fig. 4.2C), on the other hand, could be temporally extended by evolution through canalization of the morphology alone (Fig. 4.2D), since these morphologies are generally robust to the pattern of actuation. The key observation here is that only phenotypic traits that render the agent robust to changes in other traits become assimilated, a phenomenon we term differential canalization.

This insight was exposed by modeling the development of simulated robots as they interacted with a physically realistic environment. Differential canalization may be possible in disembodied agents as well, if they conform to appropriate conditions described in Supplementary Discussion.

This finding of differential canalization has important implications for the evolutionary design of artificial and embodied agents such as robots. Computational and engineered systems generally maintain a fixed form as they behave and are evaluated. However, these systems are also extremely brittle when confronted with slight changes in their internal structure, such as damage, or in their external environment such as moving onto a new terrain [27, 37, 82]. Indeed, a perennial problem in robotics and AI is finding general solutions which perform well in novel environments [117, 170]. Our results demonstrate how incorporating morphological development in the optimization of robots can reveal, through differential canalization, characters which are robust to internal changes. Robots that are robust to internal changes in their controllers may also be robust to external changes in their environment [28]. Thus, allowing robots to change their structure as they behave might facilitate evolutionary improvement of their descendants, even if these robots will be deployed with static phenotypes or in relatively unchanging environments.

These results are particularly important for the nascent field of soft robotics in which engineers cannot as easily presuppose a robot's body plan and optimize controllers for it because designing such machines manually is unintuitive [139, 188]. Our approach addresses this challenge, because differential canalization provides a mechanism whereby static yet robust soft robot morphologies may be automatically discovered using evolutionary algorithms for a given task environment. Furthermore, future soft robots could potentially alter their shape to best match the current task by selecting from previously trained and canalized forms. This change might occur pneumatically, as in Shepherd *et al.* [210], or it could modulate other material properties such as stiffness (e.g. using a muscular hydrostat).

We have shown that for canalization to occur in our developmental model, some form of paedomorphosis must also occur. However, there are at least two distinct methods by which such heterochrony can proceed: progenesis and neoteny. Progenesis could occur through mutations which move initial parameter values (ℓ, ϕ) toward their final values (ℓ^*, ϕ^*). Neoteny could instead occur through mutations which move final values (ℓ^*, ϕ^*) toward their initial values (ℓ, ϕ). Although a superior phenotype can materialize anywhere along the ontogenetic timeline, late onset mutations are less likely to be deleterious than early onset mutations. This is because our developmental model is linear in terms of process, and interfering with any step affects all temporally-downstream steps. Since the probability of a mutation being beneficial is inversely proportional to its phenotypic magnitude [76], mutational changes in the terminal stages of development require the smallest change to the developmental program. Hence, late-onset discoveries of superior traits are more likely to occur without breaking functionality at other points in ontogeny, and these traits can become canalized by evolution through progenesis: mutations which reduce the amount of ontogenetic time prior to realizing the superior trait (by moving $\ell \rightarrow \ell^*$ and/or $\phi \rightarrow \phi^*$). Indeed progenesis was observed most often in our trials (Fig. 4.6): late onset mutations which transform a walking robot into a rolling one are discovered by the evolutionary process, and are then moved back toward the birth of the robots' descendants through subsequent mutations.

Finally, we would like to note the observed phenomenon of *increased* plasticity prior to genetic assimilation. Models of the Baldwin effect usually assume that phenotypic plasticity itself does not evolve, although it has been shown how major changes in the environment can select for increased plasticity in a character that is initially canalized [128]. In our experiments however, there is no environmental change. There is also a related concept known as 'sensitive periods' of development in which an organism's phenotype is more responsive to experience [15]. Despite great interest in sensitive periods, the adaptive reasons why they have evolved are unclear [71]. In our model, increasing the amount of morphological development increases the chance of capturing an advantageous static phenotype, which can then be canalized, once

found. However, a phenotype will not realize the globally optimal solution by simply maximizing development. This would merely lengthen the *line* on which development unfolds in phenotypic hyperspace (n-dimensional real space).

The developmental model described herein is intentionally minimalistic in order to isolate the effect of morphological and neurological change in the evolutionary search for embodied agents. The simplifying assumptions necessary to do so make it difficult to assess the biological implications. For example, we model development as an open loop process and thus ignore environmental queues and sensory feedback [157, 219]. We also disregard the costs and constraints of phenotypic plasticity [165, 218]. By removing these confounding factors, we hope these results will help generate novel hypotheses about morphological development, heterochrony, modularity and evolvability in biological systems.

4.4 Methods

Ballistic development.

Ballistic development $\mathcal{B}(t)$ is simply a linear function from a starting value a to a final value b, in ontogenetic time $t \in (0, \tau)$, thus:

$$\mathcal{B}(t) = a + \frac{t(b-a)}{\tau}. \tag{4.1}$$

For Evo robots, $a = b$, hence:

$$\mathcal{B}(t) = a, \tag{4.2}$$

which is just a constant value in ontogenetic time.

Because a and b are constants set by the genotype and $\tau = 10$ (sec) is a fixed hyperparameter, development is predetermined, monotonic and irreversible—in a word: ballistic.

Current length.

For smaller voxels, it is necessary to implement damping into their actuation to avoid simulation instability. Actuation amplitude is limited by a linear damping factor $\xi(x) = \min\{1, (4x-1)/3\}$, which only affects voxels with resting length less than one centimeter, and approaches zero (no actuation) as the resting length goes to its lower bound of 0.25 cm.

Actuation of the k-th voxel $\psi_k(t)$ therefore depends on the starting and final phase offsets (ϕ_k, ϕ_k^*) for relative displacement, and on the starting and final resting lengths

(ℓ_k, ℓ_k^*) for amplitude. With maximum amplitude $A = 0.14$ cm and a fixed frequency $f = 4$ Hz, we have:

$$\psi_k(t) = A \cdot \sin\left[2\pi f t + \phi_k + \frac{t(\phi_k^* - \phi_k)}{\tau}\right] \cdot \xi\left[\ell_k + \frac{t(\ell_k^* - \ell_k)}{\tau}\right] \tag{4.3}$$

The current length of the k-th voxel at time t, denoted by $\mathcal{L}_k(t)$, is the resting length plus the offset and damped signal $\psi_k(t)$.

$$\mathcal{L}_k(t) = \ell_k + \frac{t(\ell_k^* - \ell_k)}{\tau} + \psi_k(t) \tag{4.4}$$

Current length is broken down into its constituent parts for a single voxel, under each treatment, in Supplementary Fig. 4.8.

Evolutionary algorithm.

We employed a standard evolutionary algorithm, Age-Fitness-Pareto Optimization [204], which uses the concept of Pareto dominance and an objective of age (in addition to fitness) intended to promote diversity among candidate designs. *A single evolutionary trial maintains a population of thirty robots, for ten thousand generations.* Every generation, the population is first doubled by creating modified copies of each individual in the population. The age of each individual is then incremented by one. Next, an additional random individual (with age zero) is injected into the population (which now consists of 61 robots). Finally, selection reduces the population down to its original size (30 robots) according to the two objectives of fitness (maximized) and age (minimized).

We performed the above procedure thirty times to produce *thirty independent evolutionary trials.* That the number of trials is the same as the population size within each trial is an admittedly confusing coincidence.

The mutation rate is also evolved for each voxel, independently, and slightly modified every time a genotype is copied from parent to child. These 48 independent mutation rates are initialized such that only a single voxel is mutated on average. Mutations follow a normal distribution ($\sigma_\ell = 0.75$ cm; $\sigma_\phi = \pi/2$) and are applied by first selecting what parameter types to mutate (ϕ_k, ϕ_k^*, ℓ_k, ℓ_k^*), and then choosing, for each parameter separately, which voxels to mutate. In Supplemental Materials we provide exact derivations of the expected genotypic impact of mutations, in terms of the proportions of voxels and parameters modified, for a given fixed mutation rate λ. There is a negligible difference between Evo and Evo-Devo in terms of the expected number of parent voxels modified during mutation (Supplementary Fig. 4.10).

Developmental windows.

The amount of development in a particular voxel can range from zero (in the case that starting and final values are equal) to 1.5 cm for the morphology (which ranges from 0.25 cm to 1.75 cm) and π for the controller (which ranges from $-\pi/2$ to $\pi/2$). The morphological development window, W_L, is the sum of the absolute difference of starting and final resting lengths across the robot's 48 voxels, divided by the total amount of possible morphological development.

$$W_L = \frac{1}{48(1.5)} \sum_{k=1}^{48} |\ell_k^* - \ell_k| \tag{4.5}$$

The controller development window, W_Φ, is the sum of the absolute difference of starting and final phase offsets across the robot's 48 voxels, divided by the total amount of possible controller development.

$$W_\Phi = \frac{1}{48\pi} \sum_{k=1}^{48} |\phi_k^* - \phi_k| \tag{4.6}$$

Statistical hypothesis testing.

We used the Mann-Whitney U test to calculate the p-values reported in this paper.

Data availability.

github.com/skriegman/how-devo-can-guide-evo contains the source code necessary for reproducing the results reported in this paper.

Supplementary Video S1.

youtu.be/Ee2sU-AZWC4 provides a high-level overview of the results reported in this paper.

4.5 Supplementary Discussion

Embodiment.

We consider an agent to be embodied if its output affects its input. This relationship may be represented by the simple update rule $\ell_{t+1} = f(\ell_t, \phi)$, where ℓ_t denotes the morphology of an agent at time t, and ϕ denotes its control policy. In a disembodied

system, changes to the morphology are not directly constrained by its current state; the update rule becomes: $\ell_{t+1} = f(\phi)$.

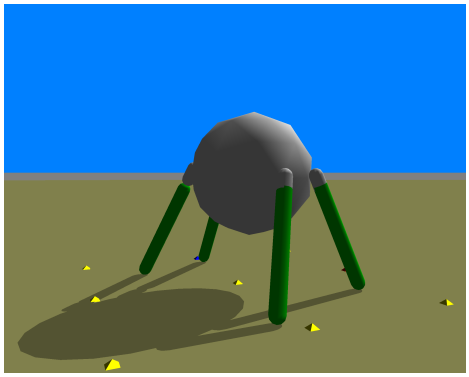
Once a round robot begins rolling, its control policy cannot instantaneously force the system to go in the other direction, since momentum will tend to preserve forward movement. This has the effect of reducing selection pressure on the controller, since fewer variations are deleterious. This allows evolution to continue climbing fitness gradients by mutating the controllers within these permissive body plans.

This might also be possible in disembodied agents if other dimensions of the system can be changed by some search process such as to facilitate the search for ϕ.

4.6 Supplementary Methods

Rigid-bodied robots.

Rigid-bodied robots and their environment were simulated using Pyrosim (ccappelle.github.io/pyrosim). The robot is a quadruped with a large, spherical abdomen; each leg is attached by a single degree-of-freedom hinge joint.



Morphological development was approximated in rigid bodies using linear actuators to slowly lengthen or shorten the length of each leg, from an evolved starting value (between 0 and 1) to an evolved final value (between 0 and 1). The controller is a simple neural net: two central pattern generators are fully connected to four motor neurons, each of which innervate a separate hinge joint. Controller development was approximated in neural networks through ballistic change to each synaptic weight: As the simulation proceeds, each weight develops linearly, from an evolved starting value (between -1 and +1) to an evolved final value (between -1 and +1).

The genotype spans two arrays: one for initial and final synaptic weights (controller), and another for initial and final leg lengths (morphology). Mutations affect, on average, a single element in each array. Apart from the genotype and its muta-

tions, the evolutionary algorithm is identical to that of the soft robots. However, the task environment now consists of a sloped floor, declined toward a light source; and performance is measured by the average light intensity recorded by a light sensor embedded in the center of the agent's abdomen, according to the inverse square law of light propagation, at each time step in its life. Occlusion of the light caused by interference of the robot's own body parts was not simulated.

The results are presented below in Supplementary Fig. S5.

Mutations for soft robots.

The following derivation shows that there is a negligible difference in the mutations produced by the Evo and Evo-Devo treatments, in terms of the number of voxels modified (Fig. S4).

Each voxel cell of a soft robot has its own material properties that can be changed by the evolutionary algorithm. Evo voxels have two material properties: (1) resting length and (2) phase offset. Evo-Devo voxels have four material properties: (1) initial resting length, (2) final resting length, (3) initial phase offset, and (4) final phase offset.

Mutations are applied by first choosing which material properties to mutate, and then choosing, separately for each property, which voxels to modify. For each of the n material properties, we select it with independent probability $p = 1/n$. If none are selected, we randomly choose one. This occurs with probability $(1-p)^n$. Hence the number of selected material properties for mutation is a random variable S which follows a truncated binomial distribution,

$$\Pr(S = s \mid n) = \begin{cases} 0 & \text{for } \; s = 0 \\ np(1-p)^{n-1} + (1-p)^n & \text{for } \; s = 1 \\ \binom{n}{s} p^s (1-p)^{n-s} & \text{for } \; s > 1 \end{cases} \tag{4.7}$$

The expected number of selected material properties is then:

$$\mathbb{E}(S) = np(1-p)^{n-1} + (1-p)^n + \sum_{s=2}^{n} s \binom{n}{s} p^s (1-p)^{n-s} \tag{4.8}$$

$$= (1-p)^n + \sum_{s=1}^{n} s \binom{n}{s} p^s (1-p)^{n-s} \tag{4.9}$$

$$= (1-p)^n + np \tag{4.10}$$

$$= (1-p)^n + 1. \tag{4.11}$$

For a selected material property, each voxel is mutated independently with probability λ, a hyperparameter we call the **mutation rate**. The expected *number* of genotype elements mutated given K total voxels is thus:

$$\delta_{\text{gene}} = \lambda K \cdot \mathbb{E}(S). \tag{4.12}$$

Dividing by the length of the genome, nK, we get the expected *proportion* of genotype elements mutated:

$$\pi_{\text{gene}} = \lambda / n \cdot \mathbb{E}(S). \tag{4.13}$$

Note that imposing bilateral symmetry does not change these expected values.

We have $K = 48$ total voxels, and $n = \{2, 4\}$ material properties for our two main experimental treatments {Evo, Evo-Devo}, respectively. The expected difference between a robot and its offspring, in terms of genotype elements, is summarized in the following table.

	$n = 2$	$n = 4$
δ_{gene}	60λ	63.8175λ
π_{gene}	0.625λ	0.3291λ

However, because multiple material properties can be mutated within a single voxel, the expected number of voxels mutated is lower than the expected number of genotype elements mutated. To calculate the average number of voxels mutated we need to consider a hierarchy of binomial distributions.

Given that S material properties were selected for mutation, the number of material properties mutated within a single voxel, M follows a binomial distribution,

$$\Pr(M = m \mid S, \lambda) = \binom{S}{m} \lambda^m (1 - \lambda)^{S-m}. \tag{4.14}$$

For brevity, let's denote the probability that at least one mutation occurs within the voxel as θ,

$$\theta = \Pr(M > 0 \mid S, \lambda) = 1 - (1 - \lambda)^S. \tag{4.15}$$

Then the number of voxels mutated, V, across a total of K voxels and S selected material properties, also follows a binomial distribution:

$$\Pr(V = v \mid S, K, \lambda, n) = \binom{K}{v} \theta^v (1 - \theta)^{K-v}. \tag{4.16}$$

And the expected number of voxels mutated (out of K total) is

$$\begin{aligned}
\delta_{\text{vox}} &= \mathbb{E}(V \mid K, \lambda, n) && (4.17)\\
&= \mathbb{E}_S \, \mathbb{E}_V(V \mid S, K, \lambda, n) && (4.18)\\
&= \mathbb{E}_S(K\theta \mid S, K, \lambda, n) && (4.19)\\
&= K \left\{ 1 - \mathbb{E}_S \left[(1-\lambda)^S \mid \lambda, n \right] \right\} && (4.20)\\
&= K \left\{ 1 - \left[(1-\lambda)(1-p)^n + \sum_{s=1}^{n} (1-\lambda)^s \binom{n}{s} p^s (1-p)^{n-s} \right] \right\} && (4.21)\\
&= K \left\{ 1 - (1-p)^n \left[\left(\frac{\lambda p - 1}{p-1} \right)^n - \lambda \right] \right\} && (4.22)
\end{aligned}$$

There is an extremely tight bound on the proportion of voxels mutated, $\pi_{\text{vox}} = \delta_{\text{vox}}/K$, for any $n > 1$ (Fig. S4). Thus mutations in Evo ($n = 2$) and Evo-Devo ($n = 4$) have practically the same impact in terms of the number of voxels modified. For completeness, the following table displays δ_{vox} for the specific values of λ considered by our hyperparameter sweep ($K = 48$) (Fig. S3).

		λ							
		1/48	2/48	4/48	8/48	16/48	24/48	32/48	48/48
n	2	1.25	2.48	4.92	9.67	18.67	27	34.67	48
	4	1.3	2.6	5.14	10.05	19.17	27.46	34.98	48

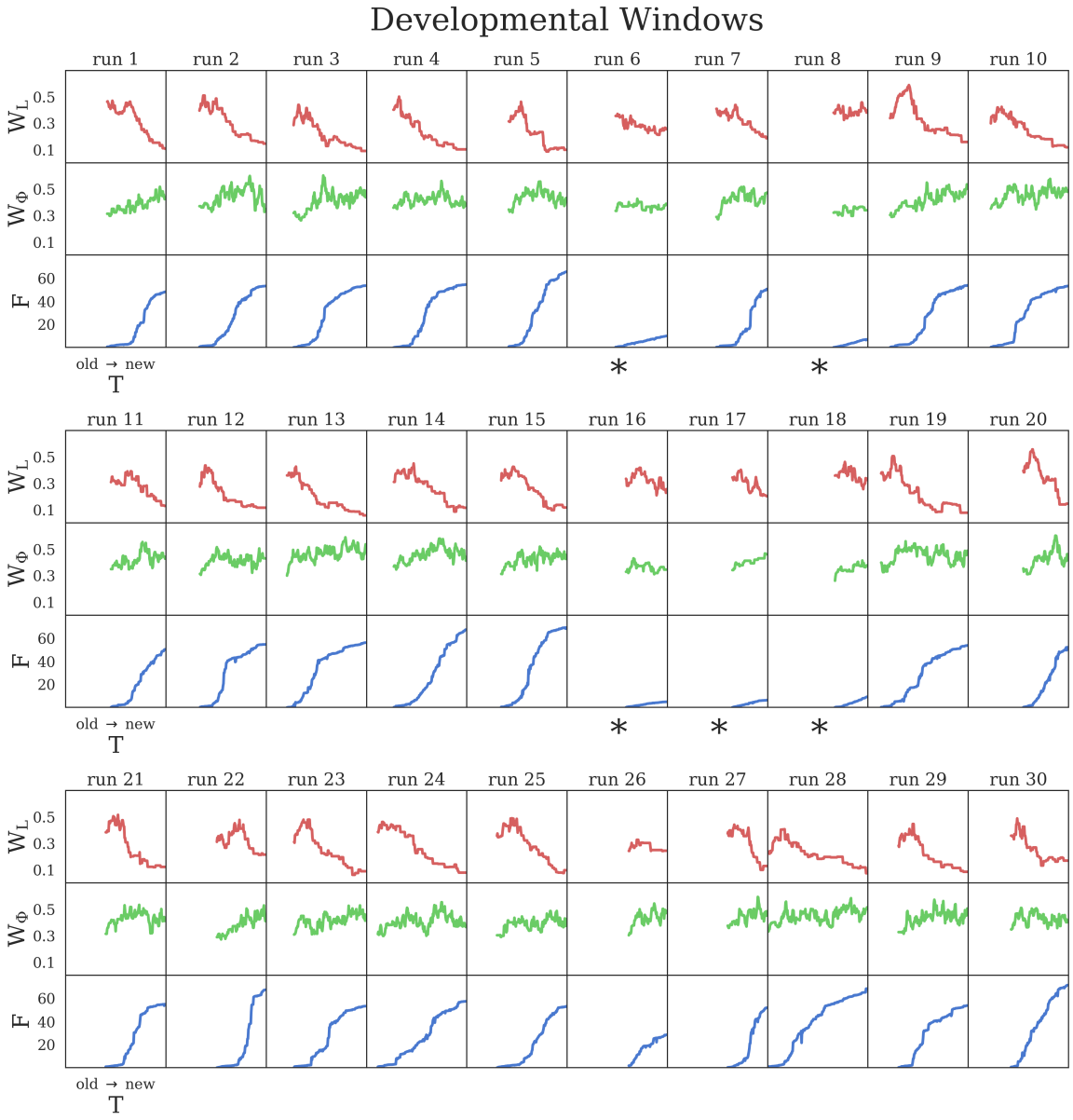


Figure 4.7: **Evolutionary change during 30 Evo-Devo trials.** The amount of morphological development, W_L (see Equation 3), controller development, W_Φ (see Equation 4), and fitness, F, for the lineages of the 30 Evo-Devo run champions. Evolutionary time, T, moves from the oldest ancestor (left) to the run champion (right). A general trend emerges wherein lineages initially increase their morphological development in T (rising red curves) and subsequently decrease morphological development to almost zero (falling red curves). Five of the 30 evolutionary trials, annotated by $*$, fell into a local optima.

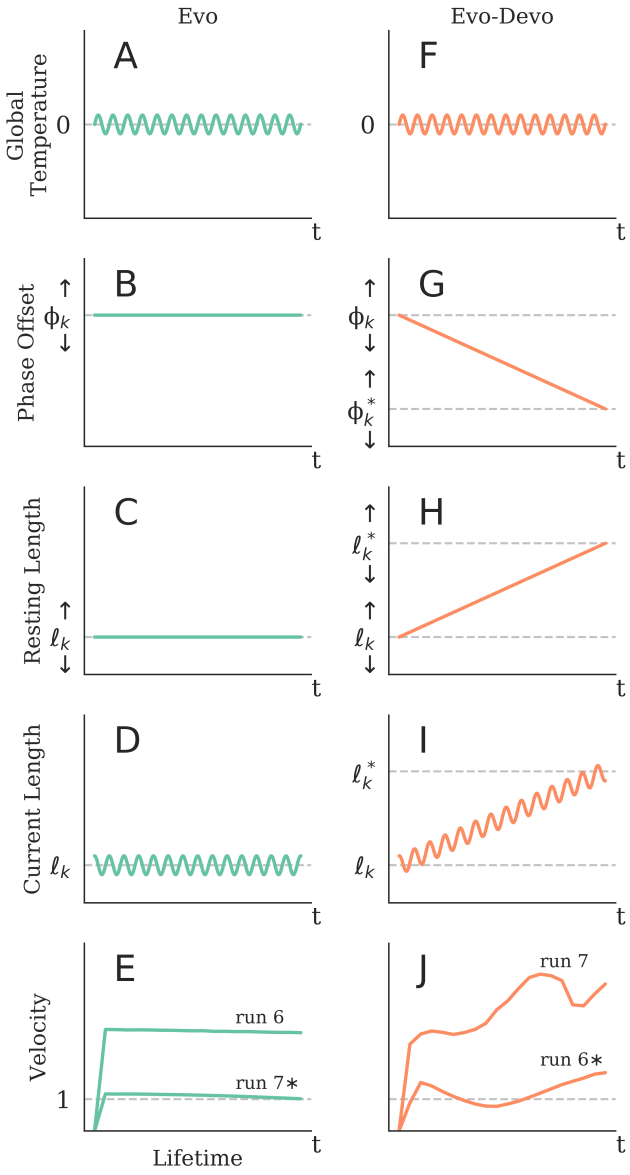


Figure 4.8: **Experimental treatments.** The phase of an oscillating global temperature (A, F) is offset in the k-th voxel by a linear function from ϕ_k to ϕ_k^* (B, G). The resting length of the k-th voxel is a linear function from ℓ_k to ℓ_k^* (C, H). For Evo, there is no development, so $\phi_k = \phi_k^*$ and $\ell_k = \ell_k^*$. The offset actuation is added on top of the resting length to give the current length of the k-th voxel (D, I). These example voxel-level changes occur across ontogenetic time (t), independently in each of the 48 voxels, and together interact with the environment to generate robot-level velocity (E, J). To see this, we averaged displacement across intervals of two actuation cycles (0.5 sec) and plotted the smoothed velocities for two Evo-Devo run champions with minimal canalization (J) alongside their non-developmental counterparts (E). Also note that the frequency of actuation is plotted here at 1.4 Hz; but, in our experiments, we used a frequency of 4 Hz.

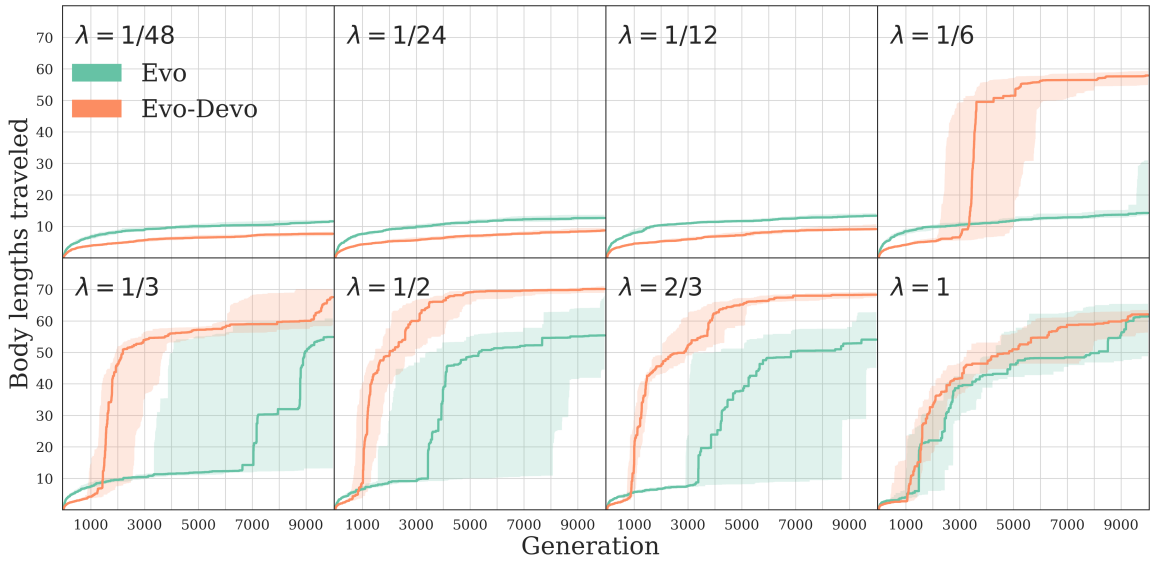


Figure 4.9: **Mutation rate sweep.** Median fitness (with 95% bootstrapped confidence intervals) under various mutation rates, λ, a hyperparameter defined in Supplementary Methods which affects the probability a voxel is mutated. In the main experiment of this paper, the mutation rate is evolved for each voxel independently, and is constantly changing. In this mutation rate sweep, λ is held uniform across all voxels. There were two observed basins of attraction in terms of fitness: a slower design that trots/gallops 5-15 body-lengths during the evaluation period, and a faster design type that rolls at 50-70 body-lengths. Although higher mutation rates facilitate the discovery of the superior phenotype, once found, lower mutation rates tend to produce more refined and faster robots. Without development, the search space has a single spike of high fitness. One can not do better than random search in such a space. When $\lambda = 1$, optimizing Evo morphologies reduces to random search, and this is the only mutation rate where Evo does not require significantly more generations than Evo-Devo to find the faster design type. This can be observed for $\lambda \in \{1/6, 1/3, 1/2, 2/3, 1\}$, by comparing the generation at which the slopes of the fitness curves increase dramatically. However, the best two treatments (Evo-Devo at $\lambda = 1/2$ and $\lambda = 2/3$), as measured by the highest median speed at the end of optimization, have development, and the robots they produced are significantly faster than those produced by random search (Evo with the highest mutation rate) ($p < 0.01$).

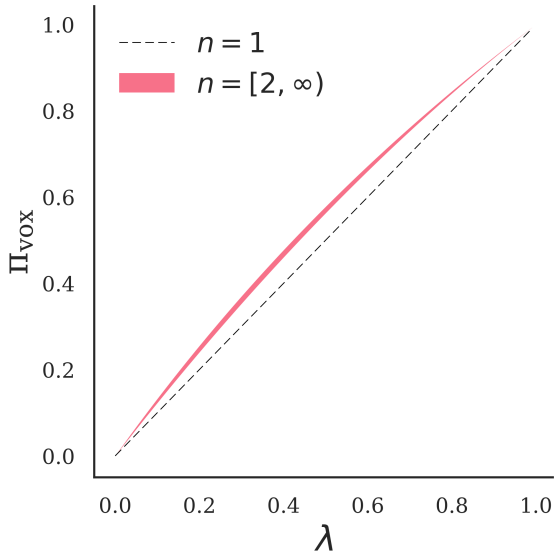


Figure 4.10: **Mutational impact.** The expected proportion of voxels modified, π_{vox}, where n is the number of material properties that can be mutated, and λ is the mutation rate. A derivation is provided in Supplementary Methods. Regardless of n, when $\lambda = 1$, every voxel must be mutated, and when $\lambda = 0$, no voxels can be mutated. Between these two points, there is an extremely tight bound on the proportion of voxels mutated for any $n > 1$. In this paper, we have treatments Evo, with $n = 2$, and Evo-Devo, with $n = 4$.

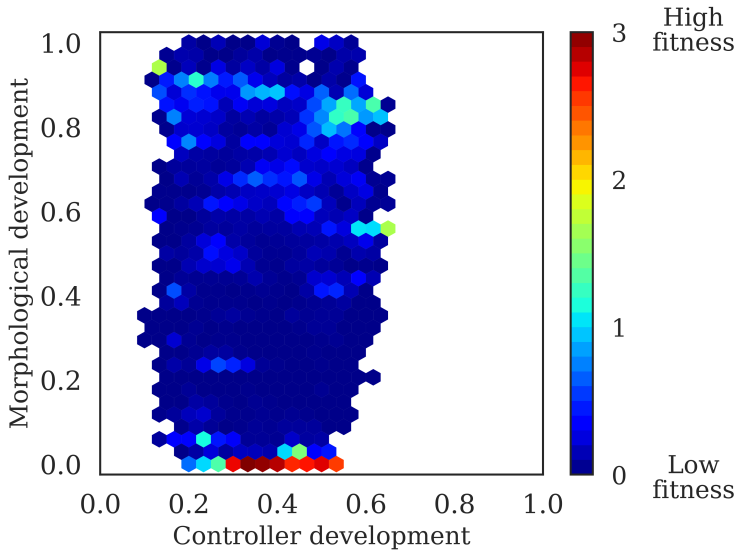


Figure 4.11: **Differential canalization in rigid bodies.** This environment consists of a sloped floor, declined toward a light source. Rigid-bodied robots, which are described in Supplementary Methods, perform phototaxis. Although longer legs produce faster walking behaviors, the shortest leg setting places the robot's large, spherical abdomen onto the ground, allowing the robot to simply roll down the ramp toward the light. An ancestor discovers rolling over toward the end of its evaluation period through ontogenetic morphological change that compresses leg lengths. This rollable morphology is assimilated to the start of development through differential canalization: The sooner a robot shrinks its legs, the sooner it can begin rolling; eventually descendants start with compressed legs, are able to immediately roll, exhibit little to no morphological change, but continue to sweep over many different synaptic weights as they behave. Rolling down the slope is not entirely passive, however, since protruding legs, even at their shortest setting, can slow down or even stop forward movement. The best controllers not only avoid such interference, but also guide rolling toward the light source. However, development in this particular case did not affect evolvability. These results are consistent with predictions made by other quantitative models used in theoretical biology that suggest that plasticity only expedites evolution under restrictive conditions [6]. We hypothesize that, in our case, this is because the search space is not deceptive enough: Once the rigid-bodied robot evolves to compress its legs, and touch its abdomen to the sloped floor, it will tend to roll for the remainder of its evaluation period. That is, in contrast with the soft robots, there is no intermediate stage between walking and rolling that suffers the fitness penalty of no longer being able to move.

Chapter 5

Material, Structure, Configuration

Appeared as:
S. Kriegman et al., Interoceptive robustness through environment-mediated morphological development. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO)* (2018).

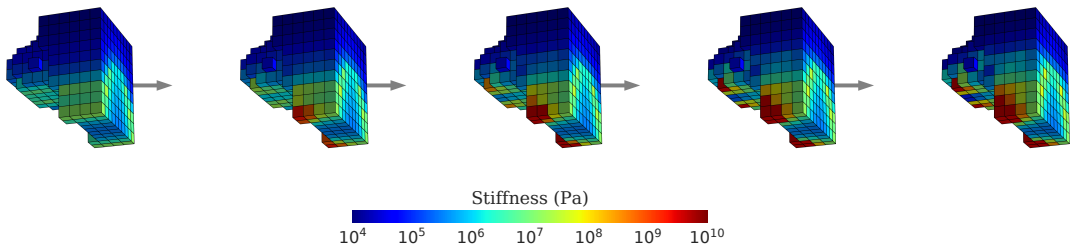


Figure 5.1: A single robot grows calluses as it walks, in response to pressure on its feet (video: **youtu.be/0cmwpcxSUWI**).

Abstract:
Typically, AI researchers and roboticists try to realize intelligent behavior in machines by tuning parameters of a predefined structure (body plan and/or neural network architecture) using evolutionary or learning algorithms. Another but not unrelated longstanding property of these systems is their brittleness to slight aberrations, as highlighted by the growing deep learning literature on adversarial examples. Here we show robustness can be achieved by evolving the geometry of soft robots, their control systems, and how their material properties develop in response to one particular interoceptive stimulus (engineering stress) during their lifetimes. By doing so we realized robots that were equally fit but more robust to extreme material defects (such as might occur during fabrication or by damage thereafter) than robots that did not develop during their lifetimes, or developed in response to a different interoceptive stimulus (pressure). This suggests that the interplay between changes in the containing systems of agents (body plan and/or neural architecture) at different temporal

scales (evolutionary and developmental) along different modalities (geometry, material properties, synaptic weights) and in response to different signals (interoceptive and external perception) all dictate those agents' abilities to evolve or learn capable and robust strategies.

5.1 Introduction

A major characteristic of life is that three broad time scales are relevant to it: evolution, development and physiological functioning. Engineered systems, in marked contrast, often employ an evolutionary or learning algorithm to improve their behavior over time, but rarely employ morphological development; any changes to the physical layout are made in between evaluations [42, 140, 215], if they are made at all.

Two notable exceptions are modular robots [257], which may reconfigure their bodies by adding and removing discrete structures, and soft robots [210], which may continuously alter the local volumes of different parts of their bodies while behaving. Others [28] have approximated topological change in rigid bodies by extending outward and angling downward appendages using a combination of linear and rotary actuators, thus simulating limb growth.

Several computational but embodied models of *prenatal* development have been reported in the literature [29, 60, 64, 68, 154]. As implied, cellular growth therein occurred prior to any physiological functioning. Thus, these studies included change during only two of the three time scales relevant to life: evolutionary and behavioral change, but not postnatal developmental change.

The most common argument in favor of development is that some aspects of the environment are unpredictable, so it is advantageous to leave some decisions up to development rather than specifying them genetically. Although self evident, it remains to determine which mechanisms of development should be instantiated in robots to realize plastic, adaptive, and useful machines.

Naturally, the performance of an evolved system depends on its capacity for evolutionary improvement: its evolvability. Development can, under certain conditions, smooth the search space evolution that operates in, thus increasing evolvability. This process, known as the Baldwin effect [12, 62], starts with an advantageous characteristic acquired during the development of individuals, such as the callouses in Fig. 5.1. This can create a new gradient in the evolutionary search space, rewarding descendants that more rapidly manifest the trait during their lifetimes [96, 122] and retain it through the remainder of their lifetime [121]. Assuming such mutations exist and can be naturally selected [122], following the gradient requires incrementally reducing development in the manifold of the search space that can express variations on the

trait [244].

However, fitness landscapes that evolution climbs, and development sometimes smooths, tend not to remain static in realistic settings. On this vacillating landscape, when the best thing to do does not remain the same, a highly evolvable but non-robust system will need to keep starting over from scratch every time the conditions change. Computational and engineered systems provide countless examples of systems with nearly perfect performance in a controlled environment, such as a factory, but who turn out to be (often comically) brittle to slight changes in their internal structure, such as damage, or their external environment such as moving on to new terrain or transferal from simulation to reality [8, 37, 85, 117, 170, 229].

Although generally absent from engineered systems (but see [27, 55]), the canonical form of robustness is seen to some extent in all organisms, and it comes from the act of development itself. For example, a plant that grows according to a fixed program will capture less light than a plant that grows toward sunlight. But there is another, more subtle form of robustness that we will refer to as 'intrinsic robustness' because it is a property of a system's structure rather than of the process by which it may change.

Developmental change produces intrinsically robust systems because they evolved from designs that had to maintain adequate performance along additional dimensions of change [28, 122]. Through morphological development specifically, evolution is compelled to maintain designs that are capable across a series of body plans, with different sensor-motor contingencies; and the ability to tolerate such perturbations can become inherited to some extent in descendants' behaviors [28] and morphologies [122], even when their developmental flexibility is reduced or completely removed by canalization or fabrication.

And yet, despite the ubiquity of morphological development in nature, and the adaptive advantages it seemingly confers, there are only a handful of cases reported in the literature in which a simulated robot's mechanical structure was allowed to change while it was behaving (e.g. [28, 114, 121, 122, 243]), all of which modeled morphological development as a genetically predetermined process: the environment could not influence the way in which development unfolded.

Assuming that an engineered system is capable of local morphological change in response to environmental signals, it is unclear how it should do so, beyond the examples of morphological plasticity observed in nature. Examples include Wolff's law [198]—bone grows in response to particular mechanical loading profiles—and Davis' law—soft tissue increases in strength in response to intermittent mechanical demands. One can envisage other such laws that are not known to occur in biology but could be helpful in a specific artificial system, such as end effectors softening in response to pressure, which might enhance their ability to safely manipulate irregular or delicate objects [34]. Indeed, the genesis of the work presented here is one such

anecdotal example given in [54], where a single robot, subjected to an abrupt doubling in gravity, stiffened its body in reaction to the increased pressure. However, whether or how it could provide a behavioral advantage, nor whether pressure is the best interoceptive signal to developmentally respond to, was not investigated.

As a step towards a more adequate picture, we introduce here a simple form of a developmental feedback mechanism: Genetic systems dictate how organisms develop in response to interoceptive stimuli, and development alters the kinds of interoceptive conditions the organism experiences. More specifically, at every time step, the proposed model of closed-loop development:

1. 'listens to' load signatures generated from movement; and, in response,

2. modifies the robot's rigidity,

which will change the way it distributes load and generates movement at the next time step.

Optimizing a system that may form a continuum of rigid and soft components—and in which this admixture may change over time—is extremely nonintuitive and underexplored. Thus, a study of the adaptive properties of such systems—and how they can best be optimized to render useful work—is initiated here.

5.2 Methods

We evolved locomotive machines constructed from voxels with heterogeneous stiffness. Like many organisms [198], the robot's material stiffness progressively changes in response to mechanical loading incurred as the robot behaves. This ontogenetic change occurs independently at each voxel according to an evolved local rule.

Physical simulation.

The soft-matter physics engine *Voxelyze* [95] is used to calculate the movement of robots resulting from their interaction with a virtual 3D terrestrial environment. Each robot is simulated for 25 times the length of an expansion/contraction cycle (a total of five seconds). The displacement between the starting coordinates and the agent's final center of mass (in the xy plane) is recorded.[1]

[1] **github.com/skriegman/2018-gecco** contains the source code necessary for reproducing the results reported in this paper.

Heavy materials.

Materials are simulated to have increased mass relative to those used by [42, 43, 94]. Constructed from heavier materials, many previously mobile robots become crushed under their own weight and require stiffer material to support locomotion with the same geometry. However, we also restricted actuation amplitude as materials grow stiffer to better approximate the properties of real materials with different stiffnesses. This creates an interesting and realistic trade-off: the stiffest material can easily support any body plan but cannot move on its own (like a skeleton without muscle), whereas the softest material can readily elicit forward movement in smaller bodies but cannot support many larger and potentially faster-moving body plans, such as those with narrow supporting limbs. Thus, a robot must carefully balance support with actuation.

Quad-CPPN encoding: $\mathbb{C}_1, \mathbb{C}_2, \mathbb{C}_3, \mathbb{C}_4$

Following [42], robot physiology is genetically encoded by a Compositional Pattern Producing Network (CPPN) [221], a scale-free mapping that biases search toward symmetrical and regular patterns which are known to facilitate locomotion.

Each point on a 10×10×10 lattice is queried by its cartesian coordinates in 3D space and its radial distance from the lattice center. An evolved CPPN takes these coordinates as input and returns a single value which is used to set some property of that point in the workspace. We used four independent CPPNs to separately encode: Geometry, Stiffness, Development, and Actuation.

$\mathbb{C}_1$ Geometry.

The geometry of a robot is specified by a bitstring that indicates whether material is present (1) or absent (0) at each lattice point in the workspace, as dictated by $\mathbb{C}_1$. The robot's geometric shape is taken to be the largest contiguous collection of present voxels.

$\mathbb{C}_2$ Stiffness.

Young's modulus is often used as an approximate measure of material stiffness. Robots are typically constructed of materials such as metals and hard plastics that have moduli in the order of $10^9 - 10^{12}$ pascals (Pa), whereas soft robots (and natural organisms) are often composed of materials with moduli in the order of $10^4 - 10^9$ Pa [199].

Here, voxels may have moduli in the range $10^4 - 10^{10}$ Pa. The robot's congenital stiffness is set at each voxel by $\mathbb{C}_2$, but may be changed by development.

$\mathbb{C}_3$ Development.

The robot's stiffness distribution k can change progressively during its lifetime t in response to localized engineering stress σ or pressure p. The rate of change α_i is specified at the i^{th} voxel by $\mathbb{C}_3$, with possible values in 0 ± 10. We compare three developmental variants.

$$\textbf{\textit{None:}} \quad \frac{dk_i}{dt} = 0 \tag{5.1}$$

$$\textbf{\textit{Stress:}} \quad \frac{dk_i}{dt} = \alpha_i \cdot \frac{d\sigma_i}{dt} \tag{5.2}$$

$$\textbf{\textit{Pressure:}} \quad \frac{dk_i}{dt} = \alpha_i \cdot \frac{dp_i}{dt} \tag{5.3}$$

$\mathbb{C}_4$ Actuation.

Robots are 'controlled by' volumetric actuation: a sinusoidal expansion/contraction of each voxel with a maximum amplitude of 50% volumetric change.[2] However, linear damping ξ is implemented into the system such that the stiffest material does not actuate (Eq. 5.5). The phase difference ϕ_i of each voxel is determined by $\mathbb{C}_4$, which offsets its oscillation relative to a central pattern generator.

Prior to actuation, each voxel has a resting length of one centimeter. This length is periodically varying ($f = 5$ Hz) by approximately 14.5% ($A \approx 0.145$ cm), but damped by ξ. The instantaneous length of the i^{th} voxel is thus:

$$\psi_i(t) = 1 + A \cdot \sin(2\pi f t + \phi_i) \cdot \xi(k_i)\,, \tag{5.4}$$

where:

$$\xi(k_i) = \frac{k_{\max} - k_i}{k_{\max} - k_{\min}}\,. \tag{5.5}$$

Evolution.

We employed a standard evolutionary algorithm, Age-Fitness-Pareto Optimization [204], which uses the concept of Pareto dominance and an objective of age (in addition to fitness) intended to promote diversity among candidate designs.

We performed twenty independent evolutionary trials with different random seeds (1-20); in each trial, a population of 24 robots was evolved for five thousand generations. Every generation, the population is first doubled by creating modified copies

[2]The scare quotes are intended to highlight the fact that actuation policies no more dictate the movement of a robot than its geometry.

of each individual in the population. The age of each individual is then incremented by one. Next, an additional random individual (with age zero) is injected into the population (which now consists of 49 robots). Finally, selection reduces the population down to its original size (24 robots) according to the two objectives of fitness (maximized) and age (minimized).

Mutations add/remove/alter a particular node/link of a CPPN. They are applied by first selecting which networks to mutate, with the possibility to select all four, and then choosing which operations to apply to each.

Hypothesis testing.

We use bootstrapping to construct hypothesis tests. All P-values are reported with Bonferroni correction for multiple (typically three) comparisons. We adopt the following convention: '$***$' for $P < 0.001$; '$**$' for $P < 0.01$; and '$*$' for $P < 0.05$.

5.3 Results

Videos of all sixty run champions (pictured in Figs. 5.2, 5.3, 5.4) can be seen at **goo.gl/T5wZNQ**.

Evolvability.

We first investigated whether environment-mediated morphological development affected evolvability (Fig. 5.5A). At the termination of an evolutionary trial, we only consider the most fit individual—the run champion—from each of the twenty independent trials (Figs. 5.2, 5.3, 5.4). After correcting for three comparisons, there was not enough evidence to reject the null hypothesis—that there is no difference between adaptive and nonadaptive robots, in terms of mean displacement—at the 0.05 level.

These results could be taken to suggest one of the following. Either the search space is sufficiently smooth prior to development (actuation and support are not as antagonistic as envisaged), or the proposed developmental mechanism is an insufficient smoothing mechanism (successful ways to change stiffness as a linear function of stimuli are sparse in the search space).

Geometric diversity.

We investigate morphological diversity next by employing the Hausdorff distance d_H as a metric to compare the similarity of two robot geometries, A and B. For each voxel in A, the closest voxel in B is identified, according to euclidean distance d.

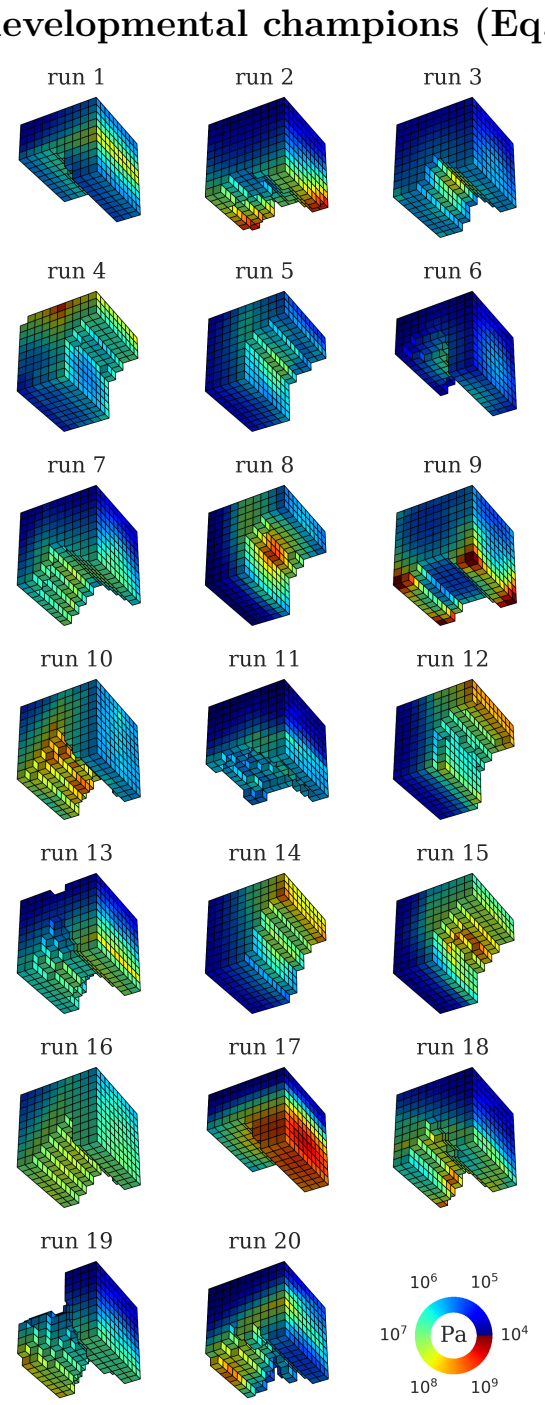


Figure 5.2: Run champions colored by congenital stiffness, which ranges from 10^4 to 10^{10} Pa. After settling under gravity, robots move toward the right-hand side of the page.

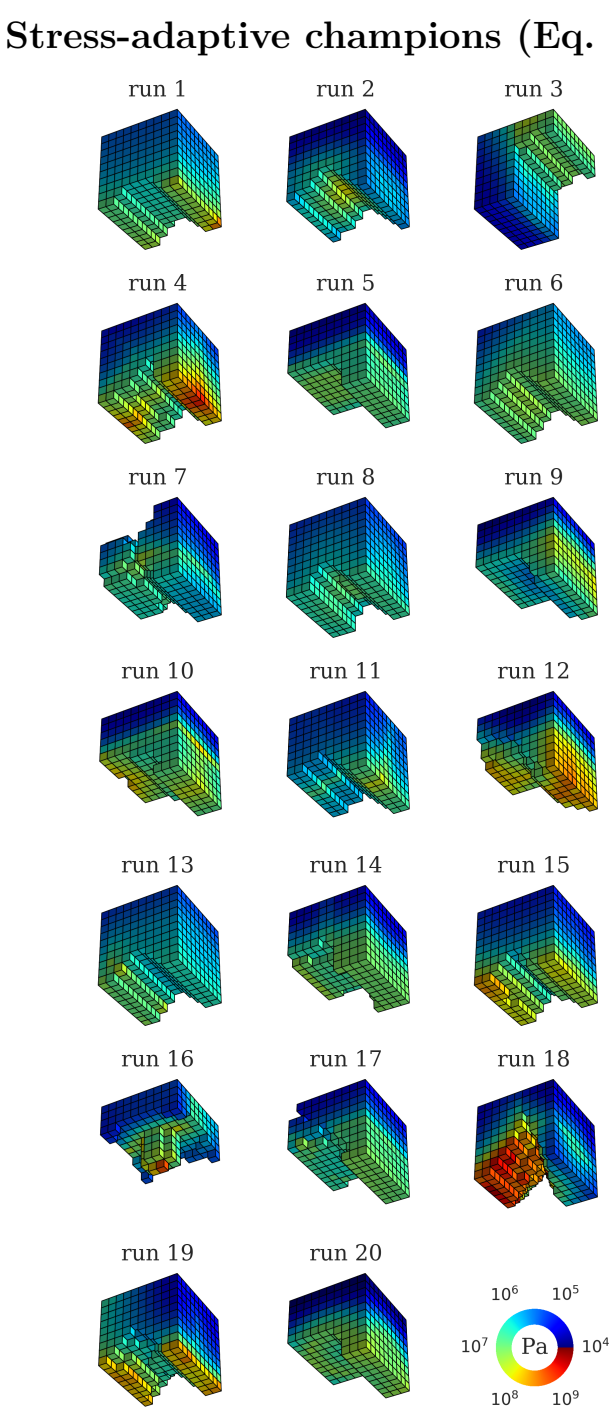


Figure 5.3: Run champions colored by congenital stiffness which can change during operation (ontogeny) in response to *engineering stress*.

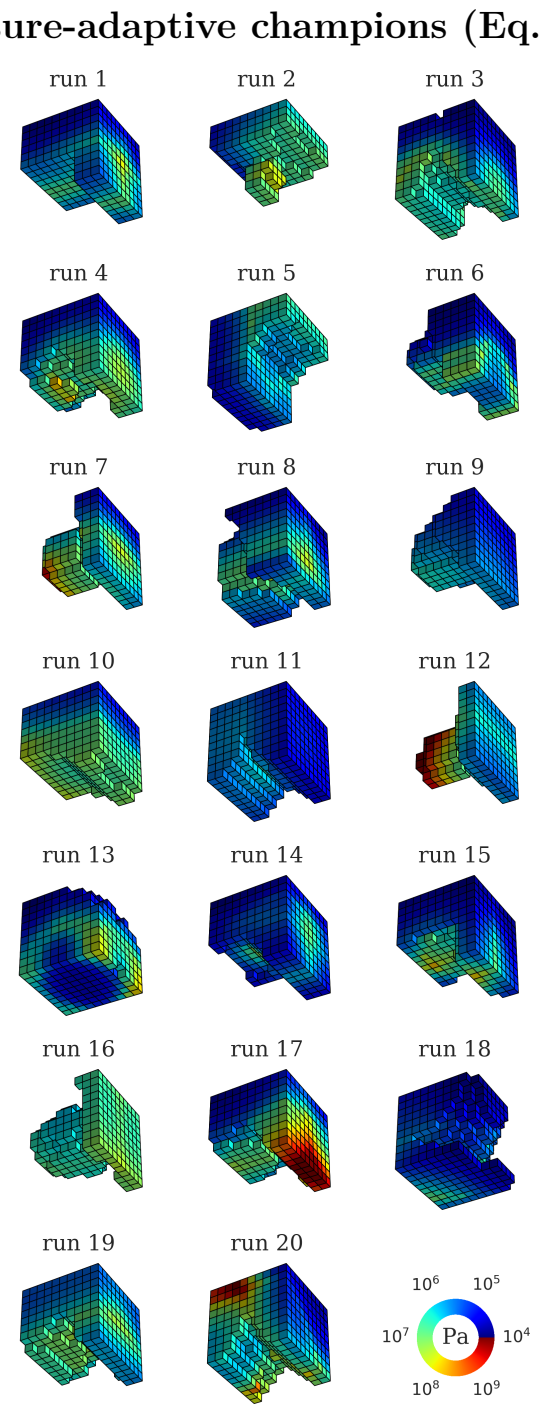


Figure 5.4: Run champions are colored by congenital stiffness which can change during operation (ontogeny) in response to *pressure*.

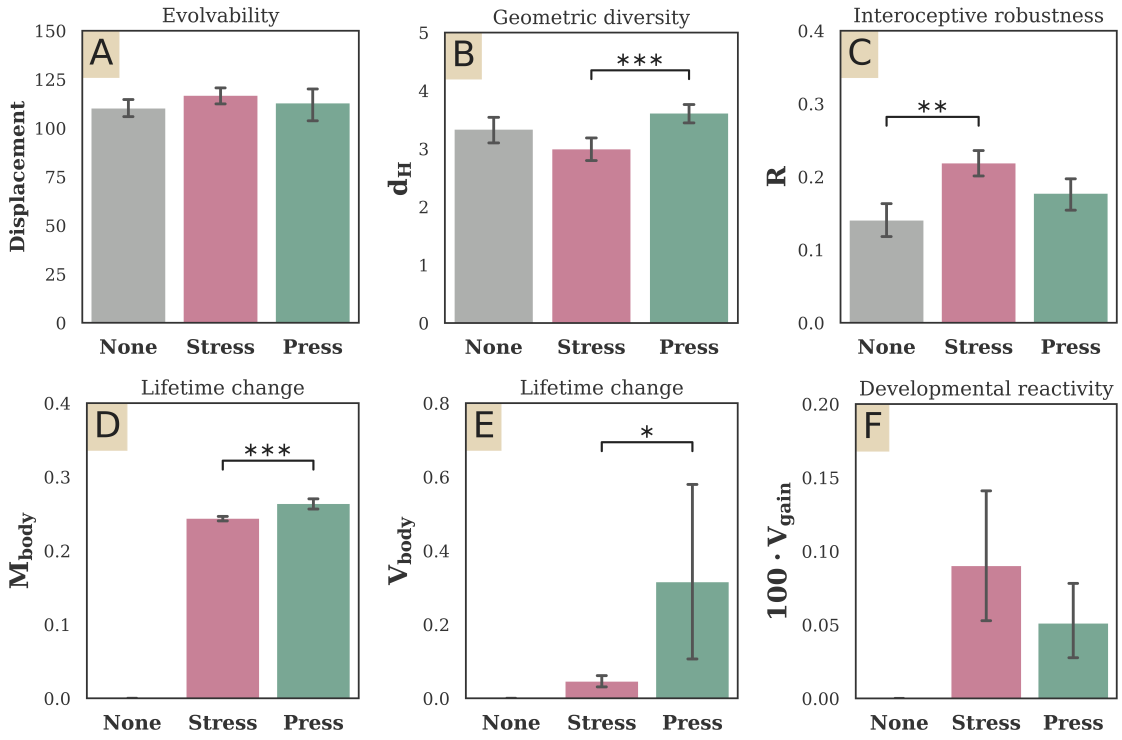


Figure 5.5: Means (with 95% C.I.) for various statistics of the run champions, at generation 5000: (A) Fitness as the final displacement of a robot, measured by voxel-length units; (B) Diversity as the pairwise Hausdorff distances of robot geometries; (C) Robustness as the relative fitness (testing fitness divided by training fitness) after development is removed and a random stiffness distribution is introduced into the champion's body (Eq. 5.7); (D) Mean, taken across the body, of relative lifetime change in stiffness, as a measure of the lack of canalization (Eq. 5.8; lower bars indicate more canalization); (E) Variance, taken across the body, of relative lifetime change in stiffness, as a measure of heterogeneity/nonuniformity in developmental reactions (Eq. 5.9); (F) Variance, taken across the body, of the coefficients/gain of developmental reactivity (Eq. 5.10).

Similarly, for each voxel in B, the closest voxel in A is identified. The Hausdorff metric is the larger of these two distances. Formally,

$$d_H(A,B) = \max\{ \sup_{a\in A} \inf_{b\in B} d(a,b),\ \sup_{b\in B} \inf_{a\in A} d(a,b) \}\,. \tag{5.6}$$

Informally, two robots are close in the Hausdorff distance if every voxel of either robot is close to some voxel of the other robot.

We calculated the Hausdorff distance between each of the $\binom{20}{2} = 190$ possible pairings of the 20 run champions (Fig. 5.5B). Because $d_H(A,B)$ depends on the orientations of A and B, we rotate B in the xy plane (0, 90, 180, and 270 degrees) and the yz plane (0 and 90 degrees), and select the rotation that creates the smallest $d_H(A,B)$.

We found the evolved body shapes of pressure-adaptive robots to be more diverse than those of stress-adaptive robots ($P < 0.001$). We did not find a significant difference, at the 0.05 level, between adaptive and nonadaptive treatments using this particular measure of morphological diversity.

Across all three treatments, there appear on visual inspection to be three types of geometries (Figs. 5.2-5.4): a Π robot with wide posterior and anterior legs; a Γ robot whose legs meet perpendicularly; and a Υ robot that connects a (mainly cylindrical) leg perpendicularly to the center of a 10×10 vertical plane. Depending on how one counts, the Υ species can be seen in at most one nonadaptive robot (Fig. 5.2, run 19), two stress-adaptive robots (Fig. 5.3, runs 7 and 16), and six pressure-adaptive robots (Fig. 5.4, runs 2, 6, 7, 9, 12, and 16). Pressure-adaptive robots have more diversity by virtue of more Υ robots.

Interoceptive robustness.

To investigate the relative robustness (if any) across the three treatments, in the following experiment, development was manually removed from the stress- and pressure-adaptive run champions. We then tested the sensitivity of the resulting reduced robots to their evolved congenital stiffness distribution (Fig. 5.5C). To do so, we replaced the evolved network dictating material stiffness, $\mathbb{C}_2$, with a random number generator that draws from the same range of possible stiffness ($10^4 - 10^{10}$ Pa). That is to say, we 'built' the evolved run champions without any errors in the specifications of geometry and actuation, but completely ignored the evolved specifications of their material stiffness, replacing them instead with random noise. We then calculated the relative fitness

$$R = F_{\text{test}} \,/\, F_{\text{train}} \,, \tag{5.7}$$

where F_{train} is the fitness achieved using the evolved stiffness and F_{test} is the fitness when tested with a random stiffness distribution. We repeated this process ten times for each run champion, each time drawing a new random stiffness distribution.

We found that, compared to nonadaptive robots, reduced stress-adaptive robots were more robust to this (extreme) discrepancy between training and testing stiffness distributions ($P < 0.01$). These results are consistent with the found correlation between development and robustness [28, 122, 154]. However, the results here indicate that this correlation is contingent on the kind of environmental signal the developing agent responds to: there was no difference between pressure-adaptive and nonadaptive robots in this regard, at the 0.05 level.

This implies that by behaving interoceptively with respect to engineering stress, robots evolved the ability to ameliorate large deviations from their expected material properties, but by behaving interoceptively with respect to pressure, robots did not evolve this character. Because development was manually removed beforehand,

robustness in our case was not a matter of changing one's body, as in the example of plant growth [226]; rather, it is an intrinsic property of structure (geometries and actuation patterns) educed from ancestors who changed in response to one particular internal state (stress), but not from those who responded to another (pressure).

The difference in robustness between nonadaptive robots and stress-adaptive, but not pressure-adaptive robots, could be due in part to the fact that there are simply more pressure-adaptive Υ robots than stress-adaptive Υ robots. While Υ robots tend to be more fit than Π and Γ robots ($P_\Pi < 0.05$; $P_\Gamma < 0.05$), they also appear to exploit their material properties to a greater degree, and are thus more sensitive to changes in its constitution, compared to Π and Γ robots ($P_\Pi < 0.05$; $P_\Gamma < 0.01$).

The Υ robot generates movement by pushing off its posterior leg, which must be rigid enough to support itself as well as propel forward the center portion of its anterior wall (e.g. run 12 in Fig. 5.4). The robot loses kinetic energy, which is stored as elastic strain energy in the spring-like voxels between the wall's center and edge. The most strain is present in the dorsal portion of the anterior wall. The springs recoil, restoring kinetic energy and generating forward motion. If the posterior leg is too soft, or the dorsal anterior wall too rigid, the Υ robot can suffer a large drop in performance.

Differences in geometry, however, shed no light on why (the reduced) stress-adaptive robots are more robust than nonadaptive robots: the level of significance ($P < 0.01$) does not change after removing the only nonadaptive robot that could possibly be classified as Υ (Fig. 5.2, run 19). Thus we continue our investigation by analyzing how stress and pressure might differentially affect the rate of developmental reactions.

Canalization.

One indication of canalization [122, 244] in our system is given by the magnitude of α_i in each voxel, as defined by Eqs. 5.2 and 5.3. However, this is but one of two necessary ingredients for a developmental reaction: it indicates bodywide responsiveness to *potential* stimuli, but ignores the *actual* stimulus.

Thus, as proxy for canalization, we measured the amount of morphological change in reaction to local stimulus, during evaluation. More precisely, we recorded the mean, across the body, of relative lifetime change in stiffness

$$M_{\text{body}} = \frac{1}{\#\gamma} \sum_{i \in \gamma} \left| k_i^+ / k_i^\circ - 1 \right|, \tag{5.8}$$

where k_i° is the congenital stiffness, k_i^+ is the final stiffness, and $\gamma = \{i : g_i = 1\}$ contains the coordinates i of each voxel g_i present in the (bit array) geometry

which has cardinality $\#\gamma$ (total voxels). Less change—lower M_{body}—indicates more canalization.

On average, voxels in stress-adaptive robots change their relative stiffness less than voxels in pressure-adaptive robots ($P < 0.001$) (Fig. 5.5D). In other words, developmental reactions are canalized to a greater extent in stress-adaptive robots. It follows, then, that the treatment with increased robustness was also the treatment with increased canalization.

To get a sense of the consistency of developmental reactions, as they occur *across the body* of evolved robots, we also recorded the spatial variance of this relative lifetime change

$$V_{\text{body}} = \text{Var}_{i\in\gamma}\left(\left|k_i^+/k_i^\circ - 1\right|\right). \tag{5.9}$$

By this measure, stress-adaptive robots exhibit more uniform reactions than pressure-adaptive robots ($P < 0.05$) (Fig. 5.5E).

Taken together, then, we may say that the developmental reactions of stress-adaptive robots are more uniform in space (lower V_{body}; Fig. 5.5E), and more canalized in magnitude (lower M_{body}; Fig. 5.5D) than those of pressure-adaptive robots. Pressure-adaptive robots therefore experience larger and more localized changes in stiffness during their lifetime.

There are two possibilities that could explain this more localized change in stiffness in the pressure-adaptive robots. One possibility is that there is greater variance among the α_i in the pressure-adaptive robots. The alternative is that there is greater variance in the application of pressure throughout the body. To test the first possibility, we first normalized α_i in pressure- and stress-adaptive robots by the differing ranges of α that evolved in the pressure-adaptive (-5.36 to 5.63) and stress-adaptive (-10.00 to 6.42) robots. Then we took the variance of α_i across the body of each run champion, individually:

$$V_{\text{gain}} = \text{Var}_{i\in\gamma}\tilde{\alpha}_i\,, \quad \text{where } \tilde{\alpha}_i = \frac{\alpha_i - \alpha_{\min}}{\alpha_{\max} - \alpha_{\min}}\,. \tag{5.10}$$

We found no evidence to support the hypothesis that α_i in pressure-adaptive robots vary more (or less) in space than those in stress-adaptive robots (Fig. 5.5F).

Therefore, because there is no difference in the variation of α_i (Fig. 5.5F), and because α_i cannot change during operation (Eqs. 5.2 and 5.3), it follows that pressure was generally much more localized within the bodies of the pressure-adaptive robots than stress within the bodies of the stress-adaptive robots. In other words, the entire body plan encountered stress, but only a small portion of the body encountered appreciable pressure. (An example of this localized response to localized pressure can be seen in Fig. 5.1.) We hypothesize that this global spread of stress is the likely cause of increased robustness in the stress-adaptive robots (Fig. 5.5C).

5.4 Discussion

Building systems that are robust in the face of changing environmental conditions is a grand challenge in robotics and AI. The brittleness of current systems is exemplified by the growing literature on adversarial examples [8, 170, 229], and the fact that almost all practical robots are confined to the perfectly flat floors they clean, or the hermetic factories built around their work. Robustness is not unknown in human-engineered systems, but it is relatively rare; in nature it is everywhere, and one of the reasons is that in nature organisms develop: They constantly change not just their cognitive architectures but the morphologies that contain them and mediate with the external world.

It has been shown for rigid robots [28] that morphological development can in some cases increase robustness since it exposes evolution to richer sensory information: the robot must maintain locomotion while changing its body. Soft robots have much greater potential in this domain: If soft, there are more ways that morphology can change, so by definition the increase in breadth in sensorimotor experiment induced by development will be even greater than that for developing yet rigid machines. Toward this goal, by allowing material stiffness to be plastic, we have here investigated a heretofore unexplored dimension of morphological change (stiffness) not available to rigid robots.

Advances in materials science and 3D printing promise new engineered systems—protean machines—that may continuously morph in response to changing environmental signals. Simply put, if a robot always changes its strategy along many morphological and neural modalities, it is more difficult to fool with a static adversarial example or a new task environment. Little to no analysis has been conducted, however, into how such systems should respond to environmental stimuli in order to adapt their functions in the face of changing environmental conditions.

In initiating such a study here, we have shown that it is not just a matter of reacting to *any* stimulus: different types of developmental feedback loops elicit different *evolved* properties. We observed that if one modality (stiffness) responds to one particular internal signal (engineering stress) but not another (pressure), robots evolved structure that intrinsically buffered large deviations from their expected material properties.

Pressure and stress bear distinct mechanical load signatures which in turn stimulated very different developmental reactions. Intriguingly, increased robustness was correlated with increased canalization: developmental reactions with stress were canalized to a greater degree than those with pressure. Although developmental reactions with pressure did not afford the evolution of robustness here, it did increase evolutionary divergence: pressure-adaptive robots evolved more diverse (congenital) shapes than stress-adaptive robots. Our work here suggests there may be other de-

velopmental feedback loops that could be made available to evolution that would lead to more diverse and robust robots.

For our purposes, 'morphology' is a robot body, but the concepts here could equally be applied to non-embodied systems, such as the architectures of deep neural networks [153, 256]. One could define internal neural processes such as node sharpening [81], Hebbian learning, or neurotransmitter diffusion [102, 241] as interoceptive signals to which the neural network developmentally responds in a structural manner, such as adding or removing neurons. Meanwhile, at a faster time scale, synaptic weights might be tuned in response to exteroceptive signals such as gradients of a loss function. Finally, such a network could be placed inside a robot which itself is experiencing morphological change.

Chapter 6

Structure, Shape, Configuration

Appeared as:
S. Kriegman et al., Automated shapeshifting for function recovery in damaged robots. In *Proceedings of Robotics: Science and Systems (RSS)* (2019).

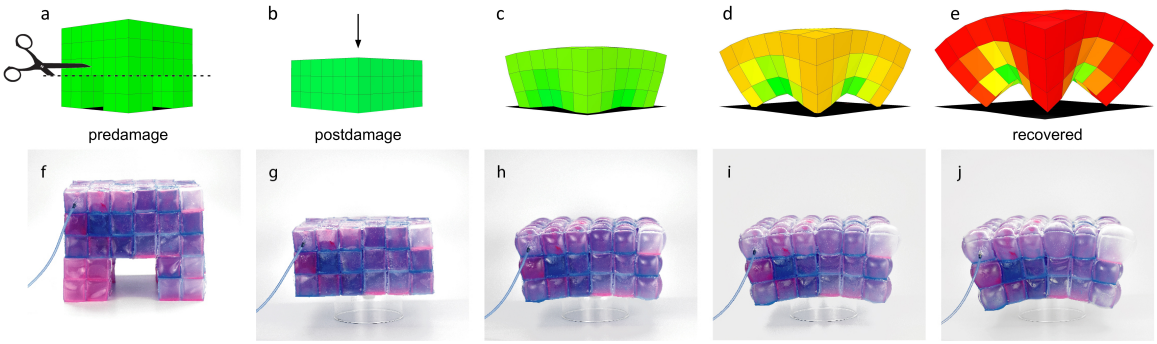


Figure 6.1: After learning to walk, a simulated quadruped is subjected to unanticipated insult: its legs are cut off. An evolutionary algorithm searches for deformations to the postdamage structure that, when coupled with the predamage controller, result in function recovery. One of the evolved solutions (shown here) yields the spontaneous "regeneration" of the lost legs, which was manually transferred to reality (**youtu.be/fFIDz8maVh0**).

Abstract:
A robot's mechanical parts routinely wear out from normal functioning and can be lost to injury. For autonomous robots operating in isolated or hostile environments, repair from a human operator is often not possible. Thus, much work has sought to automate damage recovery in robots. However, every case reported in the literature to date has accepted the damaged mechanical structure as fixed, and focused on learning new ways to control it. Here we show for the first time a robot that automatically recovers from unexpected damage by deforming its resting mechanical structure without changing its control policy. We found that, especially in the case of "deep insult", such as removal of all four of the robot's legs, the damaged machine evolves shape changes that not only recover the original level of function (locomo-

tion) as before, but can in fact surpass the original level of performance (speed). This suggests that shape change, instead of control readaptation, may be a better method to recover function after damage in some cases.

6.1 Introduction

Certain remote, hazardous or otherwise inaccessible environments preclude human intervention when a robot fails or is damaged. It would thus be advantageous for systems operating in such environments to have some capacity for self -maintenance and -repair.

Indeed, much work has investigated how, in the absence of external supervision, a robot can automatically learn new ways to control its body when damaged [27, 39, 55, 111, 127, 146, 194]. While a diverse set of recovery mechanisms have been proposed, they all shared a common assumption: The damaged mechanical structure could be reconfigured, but not fundamentally deformed.

This assumption is reasonable in classical robots, which are, generally, jointed collections of rigid links. But recent advances in materials science and 3D printing are enabling the construction of soft machines with theoretically infinite degrees of freedom and thus capable of deforming their structures so as to regenerate a lost part or embrace an entirely new geometry in the face of unanticipated insult. The possibility of such change affords a completely novel mode of damage recovery: No robot built to date has altered its resting structure in order to recover function lost due to damage.

Previous computational studies have demonstrated structural but non-functional change in discrete models. For example, cellular automata have been trained to grow a target structure from a single cell [68, 154]. Similar growth rules could in principle be instantiated in self-assembling modular robots [251, 257]. However, structural change would require access to additional modules in the environment, redundant modules on the body, or the ability to internally generate them. Moreover, it is unclear how or if such rules could dictate continuous geometric deformation in soft robots.

The present work builds on two closely-related research projects in which injured robots automatically generate and test candidate control policies in order to find compensatory behaviors that work in spite of damage [27, 55].

In the first, Bongard et al. [27] demonstrated how, under the right conditions, an autonomous robot could internally model its own geometry with minimal sensorimotor experiment. The benefit of this approach is that, once a sufficiently accurate self-model has been established, actions can be internally rehearsed, discarding those which are unsuccessful or dangerous, before attempting them in reality. If model accuracy drops, as from structural changes due to damage, modeling resumes and

continues until the robot's current morphology is adequately reflected in the robot's model of self.

The main drawback of this approach is that internal modeling requires additional computation, and there are circumstances in which the robot cannot afford—in terms of time, money, energy, stability, and the overall well-being of itself and others—to remain stationary for extended periods of time.

To speed recovery, Cully et al. [55] proposed that robots should instead exploit the fact that resources prior to deployment are relatively cheap in terms of the factors listed above. A large, behavioral repertoire composed of mappings from behaviors (for the undamaged robot) to their predicted performances can therefore be modeled in simulation beforehand, and come preinstalled on the robot. Assuming damage is detected by an external mechanism, the authors showed how, under certain conditions, such a map can be rapidly updated and traversed to find successful behavior, which is implicitly robust to differences between the current and pre-deployment morphologies.

The robots used in this past work consisted of rigid components attached together with a handful of mechanical degrees of freedom: The quadruped in [27] had 8 motors and 4 DOF; the hexapod in [55] had 18 motors and 12 DOF. The control problem was greatly impacted by these mechanical details and their intrinsic dynamics, but they were taken as given, even when damaged, because these robots simply could not deform their resting structure.

Instead of treating the body as just the problem domain, we here modify it as part of the computational loop. This is possible because our robot has many more (140) mechanical degrees of freedom, and the ability to change the volume, rather than just the relative displacement, of each component. This flexibility enables a heretofore unexplored mode of damage recovery: keep the existing controller but deform the resting structure. Existing approaches to controller adaptation could in principle (although this is not investigated here) be paired with such changes to morphology. However, in many cases, it would be desirable to retain a previously optimized and fine-tuned controller, especially if missing structure can simply be regenerated.

We here show that, under a wide range of damage scenarios, automated shapeshifting can be advantageous, and that, in most of the cases tested, shapeshifting alone (holding the existing controller fixed) outperforms controller adaptation alone (holding the damaged shape fixed), in terms of recovered mobility.

6.2 Methods

This section describes the hardware, simulation and control of our robot, the damage scenarios it faces and its options for recovery: shapeshifting and controller adaptation.

We also define a tripartite classification—of 'structure', 'shape' and 'configuration'—that forms the basis of our argument, which is, briefly, that the way in which our robot recovers from damage—shape change—was outside the scope of any robot previously reported in the literature.

The source code.

github.com/skriegman/2019-RSS

The robot.

Figure 6.2: The blue robot, made from thin-walled inflatable elastomer voxels.

The robot is an isobilaterally symmetrical quadruped constructed from 140 inflatable silicone "voxels" (Figs. 6.1f and 6.2). We here present a method for creating air-filled voxel membranes with relatively uniform thickness.

Creating thin, hollow 3D silicone structures is challenging due to several factors, including mold precision and potential for damage during release from molds. One effective but labor-intensive method is to make the 3D shapes by adhering 2D films at their joints [162]. Here, inspired by a scalable 2-axis rotational molding technique [255], we employ a 1-axis rotational drip-molding machine.

First, silicone (Dragon Skin 10 Fast; Smooth-On, Inc.) was poured into an open-face acrylic mold and a tongue depressor was used to roughly spread the silicone along the walls. The mold was then attached to the rotational molding machine with the rotation axis oriented downward at 45 degrees relative to horizontal, and run through cycles comprising a 90° turn, stopping for 45 seconds after each turn to allow the silicone to flow and evenly coat each side. Excess silicone dripped out of the mold, leaving a thickness which was dependent on several interrelated factors including the cure time, viscosity, and the interaction between the silicone and acrylic.

After the silicone cured, excess material was cut away. A silicone base-layer was then rod-coated onto a flat acrylic sheet. Next, the bottomless cubes were placed on the base-layer and allowed to cure, sealing air inside each voxel. The voxels were then cut from the sheet and a small hole was punched in each voxel for tubing. Finally, silicone tubes were inserted and bonded with Sil-Poxy (Smooth-On, Inc.).

The overall robot consists of a $6 \times 6 \times 3$ voxel torso and four removable $2 \times 2 \times 2$ voxel legs (Figs. 6.1f-j and 6.2). Sil-Poxy and Ecoflex 00-50 were used to improve adhesion between voxels. To explore the effect of layer thickness on the range of attainable morphologies, two versions of the robot were fabricated: The **blue robot**

(Fig. 6.2) consists of voxels made with one layer of silicone, while the **purple robot** (Fig. 6.1f-j) consists of thicker-walled voxels made with two layers of silicone.

Individual cubic voxels were manually inflated at pressures less than 20 kPa, and approached a spherical shape as pressure increased. When patterned together into a robot, selective inflation of a subset of voxels induces overall robot shape change. To reduce friction and weight effects in the robots, they were placed on top of a glass crystallizing dish, which lifted their legs off the table surface. While this arrangement made motion difficult, it allowed us to conduct a preliminary investigation of the feasibility of transferring simulated shape change to a physical system. In future implementations, the manual inflation could be replaced by pressure regulators [31], allowing the robot to approach the continuous control achievable in simulation.

To understand some of the trade-offs between design parameters, consider a spherical pressure vessel in uniform free expansion:

$$p = \frac{2E \cdot \epsilon \cdot t}{r} = \frac{2E \cdot \epsilon \cdot t_0 \cdot (1 - \delta)}{r_0 - \epsilon}, \tag{6.1}$$

where t_0 [m] is the thickness of the pressure vessel, r_0 [m] is the radius, ϵ is the linear strain due to expansion, E [MPa] is Young's modulus, and δ is the radial strain (which is determined from ϵ and the material's Poisson's ratio). Note that each voxel can push outward with a force proportional to the pressure. Examining Eq. 6.1, we see that at a given strain rate and initial dimensions, the internal pressure scales linearly with both thickness and modulus. Thus, when choosing thickness of voxels, there was a tradeoff between weight and internal pressure: doubling the wall thickness doubled weight, in exchange for doubled operational pressure.

The simulation.

To simulate the robot, we use the voxel-based physics engine *Voxelyze* [95], which simulates elastic voxels using two elements: particles and beams. Particles have mass and rotational inertia, and are connected on a cartesian grid by spring-like beams (with translational and rotational stiffness). For visualization and reference, part of a voxel mesh is drawn around this structure such that each voxel has a single particle at its center (Fig. 6.3).

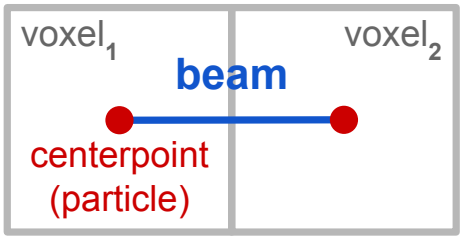


Figure 6.3: Voxels are simulated by beams (springs) and particles (masses).

Two adjacent voxels are connected, centerpoint to centerpoint (i.e., particle to particle), by a single, shared beam. Material properties (e.g., volume and elasticity) are specified at the particles but implemented as attributes of beams (e.g., their rest

length, and how easily they twist and stretch). Where two adjacent particles disagree in their "desired" attributes of a shared beam, an average is taken.

A beam exits a voxel normal to, and in the center of, one of the voxel's faces. Although the mesh is drawn such that voxel edges bend around the underlying beam-mass network (see, e.g., Fig. 6.1), a spherical envelope is used for collision detection, thus approximating the spherical expansion of the physical voxels (with maximal expansion occurring at the center of each face). For more details see [95].

The structure and shape of a robot.

The **structure**, $\mathbb{S}$, of a robot is determined by the number and placement of voxels, and simulated by the presence and absence of particles on a regular grid in the workspace. Let the bit value v_i denote the presence ($v_i = 1$) or absence ($v_i = 0$) of a voxel at index i. The structure,

$$\mathbb{S} = \{i : v_i = 1\}, \tag{6.2}$$

is thus a set of voxel coordinates.

The **shape**, $\mathcal{S}$, of a robot is determined by the resting volume of each voxel, which is expressed in simulation as the resting (or, equilibrium) lengths of the beams connecting adjacent particles, and in reality as a resting pressure within each voxel (though the exact pressure, p_i, is not measured here). Let the floating point value b_i denote the beam rest length stored at the i-th simulated voxel. The shape,

$$\mathcal{S}_{\text{sim}} = \{b_i : i \in \mathbb{S}\} \ \sim \ \mathcal{S}_{\text{real}} = \{p_i : i \in \mathbb{S}\}, \tag{6.3}$$

is thus a set of voxel resting volumes.

The robot has a quadrupedal predamage structure (Figs. 6.1a,f and 6.2) with atmospheric voxel resting pressure, which is approximated by nominal beam rest lengths of 1 cm. Damage removes structure (voxels) (Fig. 6.1b). Postdamage structural deformation—shape change—is executed by pressure changes in the remnant structure (i.e., mutations in $\mathcal{S}_{\text{real}}$) (Fig. 6.1h-j) and approximated by local adjustments in the remaining beam-mass network (i.e., mutations in $\mathcal{S}_{\text{sim}}$) (Fig. 6.1c-e). The mechanical structure and its resting shape are fixed prior to behavior during the evaluation period (20 actuation cycles).

The controller and configuration of a robot.

The controllers continuously reconfigure the volume of a given mechanical structure during the evaluation period. We here consider open loop control of ± 0.5 cm^3 volumetric change ($\pm 50\%$ from nominal), at each voxel, with a phase offset relative to a central pattern generator, for 4 sec.

Controllers are here encoded as neural networks that map the indices of voxels in 3D space (Eq. 6.2) to a phase offset value, ϕ_i, between -2π and 2π. We chose this particular encoding, which is commonly referred to as a Compositional Pattern-Producing Network, or CPPN [221], because spatial regularities (in structure and actuation) are known to facilitate locomotion. (For more details about this encoding, see [42].)

The instantaneous **configuration**, $\mathcal{C}$, of a robot is determined by an oscillating adjustment to the volume (and thus pressure) of each voxel, centered around its shape $\mathcal{S}$. In simulation, rest lengths are periodically varying ($f = 5$ Hz) around their baseline, b_i , with constant amplitude ($A \approx 0.145$ cm), but damped by ξ. Damping prevents contracting voxels from overlapping by decreasing their oscillation amplitude as their rest length approaches a lower bound of $b_i = 0.25$ cm.

The instantaneous adjustment to the rest length of the i-th simulated voxel, at time t, is thus:

$$\psi_i(t) = A \cdot \sin(2\pi f t + \phi_i) \cdot \xi(b_i), \tag{6.4}$$

where:

$$\xi(b) = \min\left[1,\ \frac{4b-1}{3}\right]. \tag{6.5}$$

The configuration,

$$\mathcal{C}_{\text{sim}}(t) = \{b_i + \psi_i(t) : i \in \mathbb{S}\}, \tag{6.6}$$

is thus a cyclical adjustment in the rest length between adjacent simulated voxels (implemented when computing the elastic force between them) throughout a structure $\mathbb{S}$ with shape $\mathcal{S}_{\text{sim}}$.

Although simple, open loop control has the ability to produce complex behaviors, such as symmetrical and asymmetrical gaits (from patches of voxels that oscillate in counter-phase), or propagating waves of excitation (from a sequence of voxels with increasing or decreasing phase offsets). Indeed, it is well known that central pattern generators in the mammalian spinal cord (and elsewhere in invertebrate systems) produce the basic, rhythmic motor patterns of locomotion, such as stepping, independently of sensory input [86].

The damage scenarios.

We here consider nine damage scenarios—we amputate: (i) half of a leg; (ii) one entire leg, (iii) two adjacent legs, (iv) two diagonal legs, (v) three legs, (vi) all four legs; (vii) one quarter of the robot's body, (viii) one half of the body, and (ix) three quarters of the body.

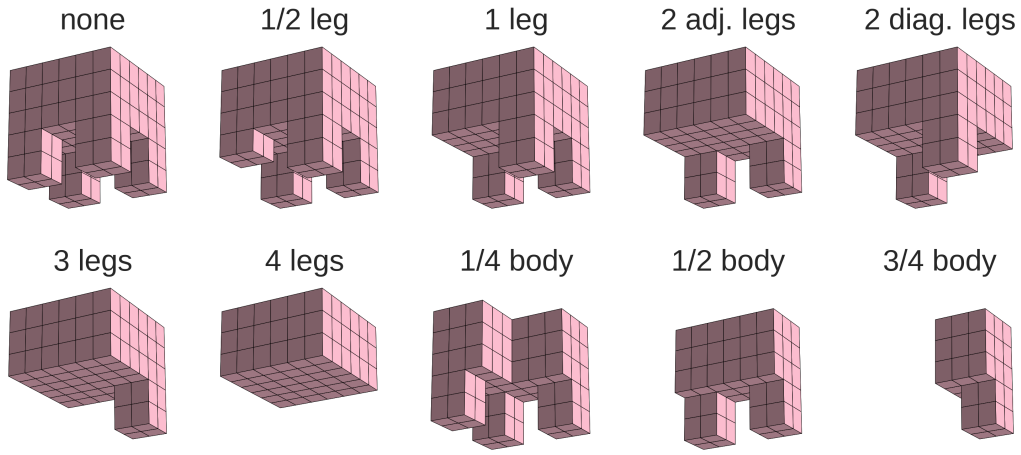


Figure 6.4: The various amputations applied in our experiments. The predamage robot (amputation = 'none') is shown for reference.

The recovery options.

Each damage scenario removes structure and breaks the robot's functionality: the robot loses voxels and its ability to walk. We here consider two options for function recovery:

1. **Controller readaptation.** A new controller is optimized for locomotion with the damaged structure, as in [27, 55]. The only parameter subject to (re)optimization is the phase offset, ϕ_i, of each voxel.

2. **Shapeshifting.** The shape of the damaged structure is optimized for locomotion with the existing controller. The only parameter subject to optimization is the baseline rest length, b_i, of each voxel.

The shape change.

The body is reshaped *prior to* behavior (i.e., before the controller is turned on), analogous to a prenatal developmental stage. This is done by adjusting the robot's shape, $\mathcal{S}$, as defined in Eq. 6.3. Then, behavior results from oscillations that are symmetrically distributed about this shape (Eq. 6.6).

The same kind of neural network that encodes controllers (i.e, a CPPN) was also used to encode the robot's shape. However, the shape-encoding networks output a rest beam length, b_i, between 0.25 and 2 cm (instead of a phase offset, ϕ_i, between -2π and 2π). Subject to the constraints outlined above, optimization searches for shape-encoding networks that result in resting shapes that, when coupled with the original

open-loop controller (previously optimized for the undamaged robot), synergize to recover forward movement.

The optimization algorithm.

Shapes and control policies are here optimized to displace the (simulated) robot in any direction using Age-Fitness-Pareto Optimization [204], an evolutionary algorithm that uses the concept of Pareto dominance and an objective of 'age' (in addition to displacement) intended to promote diversity among candidate designs and prevent premature convergence.

A trial is initialized with a population of 50 randomly-generated designs with age zero. Every generation, the population is first doubled by creating modified copies of each individual in the population (i.e., offspring, in which 'age' is set equal to that of the parent), where modification occurs only to the encoding-network that is currently being optimized (either that of ϕ or b). The age of each individual is then incremented by one. Next, an additional random individual (with age zero) is injected into the population (which now consists of 101 designs). Finally, selection reduces the population to its original size (50 designs) according to the two objectives of net displacement (maximized) and age (minimized): Starting with nondominated designs ($N = 0$), successive Pareto fronts (containing designs dominated by exactly N alternatives, for $N = 1, 2, \ldots$) are kept in their entirety until doing so would overfill the population past its original size; then, designs are selected one-by-one with probability proportional to their net displacement. (The 51 unselected designs are deleted.)

This process of random variation and directed selection is repeated for G generations, in which both the architectures and weights of the encoding networks are optimized: Mutations add, modify or remove a particular vertex or edge. Where modification of an edge reweights it (within -1 to 1 bounds) by adding a value randomly drawn from a normal distribution with mean zero and standard deviation 0.5. Vertex modification swaps the node's activation function with a randomly chosen function in the set (adopted from [123]): sin(), abs(), square(), sqrt(abs()); and the negations of those four.

6.3 Results

Prior to damage, twenty controllers were optimized (for $G = 1500$ generations) to generate forward movement in the simulated quadruped during an evaluation period of 4 sec (with numerical integration time steps of 0.000151 sec). Predamage displacement ranged from 37 to 46 cm (6.2 - 7.7 body lengths).

In order to isolate the effect of shape change relative to that of controller adaptation, across a diversity of insult, the simulated robot is copied $9 * 2 * 20 = 360$ times; once for each unique damage scenario (9 total), recovery option (2 total) and controller (20 total) triplet. Each copy is thus given an optimized controller and recovery option, cut according to its particular damage case, and then reoptimized for displacement (for $G = 500$ postdamage generations).

The performance recovered after damage.

Figure 6.5 plots mean relative performance (i.e., postdamage displacement as a fraction of predamage displacement), with 99% bootstrapped confidence intervals, for the two recovery options in each damage scenario.

There are two, *independent* recovery options: controller readaptation (control) and shapeshifting (treatment). For each damage scenario, the data consist of two random samples, a sample from the control population (20 independent trials, holding the shape of the damaged structure fixed) and an independent sample from the treatment population (20 independent shape optimization trials, holding the controller fixed). On the basis of these samples we wish to investigate the presence of a treatment effect that results in a shift of location (median). The null hypothesis is that of no treatment effect; the samples can be thought of as a single sample from one population.

We used a distribution-free rank sum test (Wilcoxon, Mann and Whitney) for the hypothesis of no treatment effect, with Bonferroni correction for nine comparisons. The corrected rank sum test and the 99% bootstrapped CIs (of the mean) are in agreement. That is, statistical significance between shapeshifting and controller adaptation, at the 0.01 level, can be correctly inferred, for each damage scenario, by visual inspection of Fig. 6.5 (i.e., no overlap in the shaded confidence intervals here implies rejection of the null hypothesis).

Overall, shape change was more successful (often better and never worse) than controller adaptation. Interestingly, the proportion of fitness recovered was in some cases higher than one. This could be due to a lack of volume conservation and the possibility that larger robots simply run faster than small ones. However, many robots recovered by reducing their overall volume (e.g., Figs. 6.7 and 6.8). Moreover, controller adaptation also achieved higher-than-predamage performance in one case (amp. = 2 adj. legs), and this phenomenon was also documented in [55] but not via shape change.

To explicitly control for the effect of body size, we optimized the controller of an otherwise identical quadruped that is twice the size of the original (Fig. 6.6). Isometrically increasing volume did not affect speed: There was no significant difference in speed (at the 0.01 level) between the enlarged and original quadruped. This is because the controller oscillations are added on top of (not relative to) the root

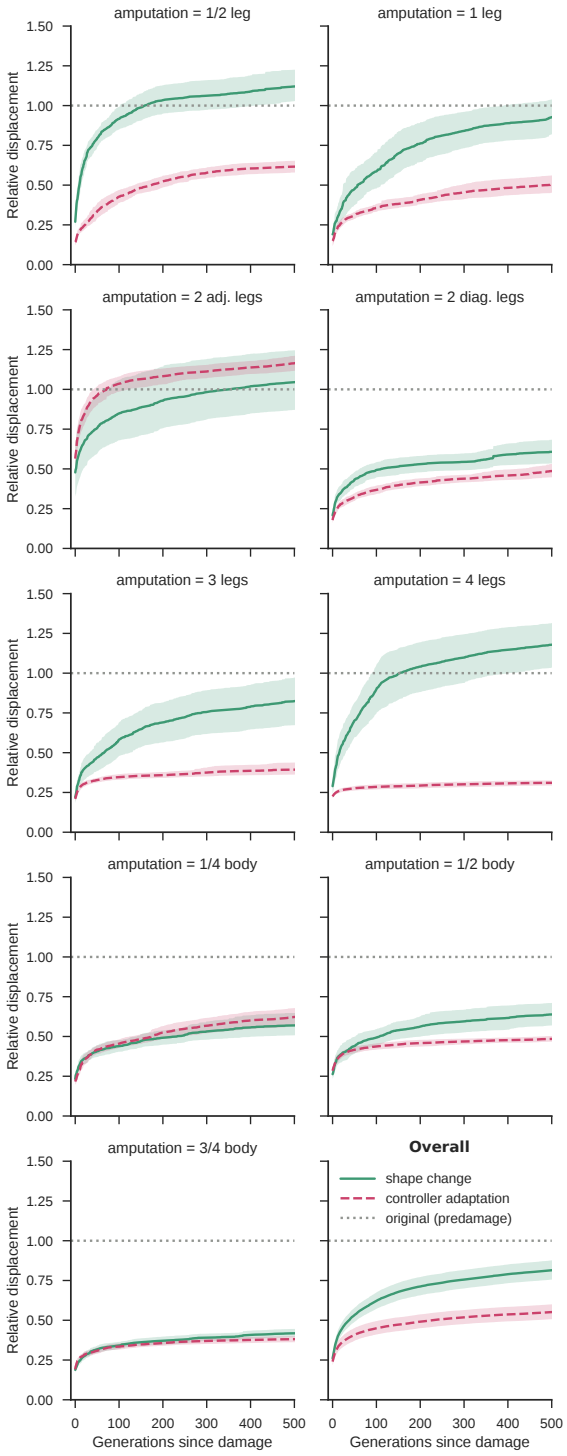


Figure 6.5: Mean relative displacement (i.e., recovered performance) with 99% CIs, at each generation (g) of reoptimization since damage occurred $(g = 0)$.

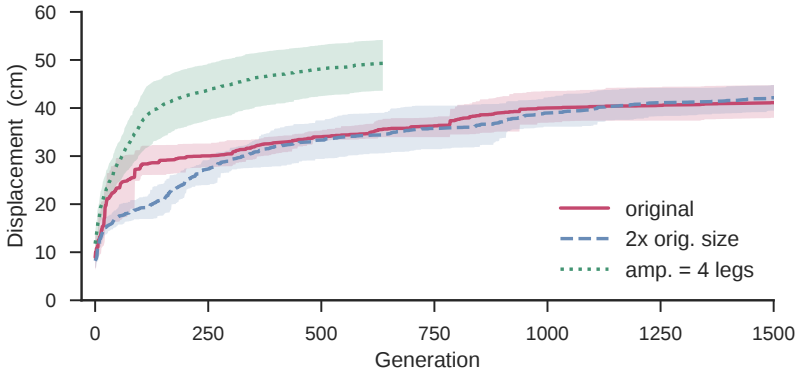


Figure 6.6: Mean displacement with 99% CIs of controller optimization in the predamage quadruped isometrically enlarged to maximal volume, compared to shape optimization after amputation of all four legs.

shape (Eq. 6.6). The enlarged robot has eight times the volume of the original, beam length oscillations still have the same amplitude.

Despite the fact that control was optimized for the original quadruped, and that amputation of all four legs removes 23% of the original volume and actuation (Fig. 6.4), robots that recovered from this particular insult through shape change (Fig. 6.1) move significantly faster than both the original and isometrically enlarged quadrupeds (Fig. 6.6). It follows that the efficacy of shapeshifting is not due simply to increased volume; rather, it is due to where and how the remnant structure's shape is deformed, which affects (e.g.) the robot's posture and mass distribution, its points of contact with the ground, and the storage and release of elastic strain energy, during locomotion.

The found techniques of recovery.

We found that the optimizer discovered diverse recovery strategies through shape change (Figs. 6.7-6.9), whereas controller readaptation often converged on the same strategy. For example, with all four legs removed due to damage, the robot is reduced to a cuboid, and the only viable locomotion technique found by controller readaptation was crawling. During postdamage optimization, many of these robots evolved crawling by peristalsis or in a manner that resembles the serpentine crawling of snakes [4].

Deforming the structure, even in a random manner, tends to produce greater frictional anisotropy which enhances peristalsis and serpentine crawling, and enables yet simpler forms of movement such as two-anchor crawling [4].

Nevertheless, crawling is inefficient because of drag. Here, in many damage cases, behavioral competence was recovered through shape changes that partially or com-

pletely (but, due to material constraints, never perfectly) regenerated missing legs. Notably, when all four legs were amputated, recovery strongly converged on the solution of regeneration, and the resulting designs were some of the fastest overall (Fig. 6.1). Note that the objective function does not assume or directly select for legged locomotion.

Other amputations, however, can be beneficial, if they result in a shape that is easier to efficiently control than the original: Prior to damage, the robot's sagittal silhouette resembles a Π. When two adjacent legs are amputated, the resulting Γ shape, which initially falls forward like this ↶ due to gravity, tends to rapidly surpass predamage performance through controller readaptation alone, despite its diminished size.

When only half of a leg was lost to injury, some robots contracted all of the undamaged legs to recover a stable but shorter quadrupedal form. Others seemed to simply regenerate the missing part through local volumetric expansion at the site of damage: The stump was isometrically expanded into a leg that was the same length as the original but much wider.

On closer inspection, however, many of those who regenerated a limb also made various other compensatory shape changes away from the site of damage, such as expanding and curving their spine. Thus even when damage is isolated to a small part of the robot's structure, global changes, in addition to local repairs, can sometimes streamline recovery.

When damage was distributed across a wider portion of the body, a diversity of solutions were discovered. For example, after the amputation of a quarter of the robot's body, the robot occasionally splayed out its pelvis to form a straighter and faster shape (Fig. 6.7). And, after the amputation of three legs, some robots once again grew replacements, but, because of other changes (e.g., a greatly expanded back), the remaining "genuine" leg needed to be partially contracted and tucked inward for balance (Fig. 6.10).

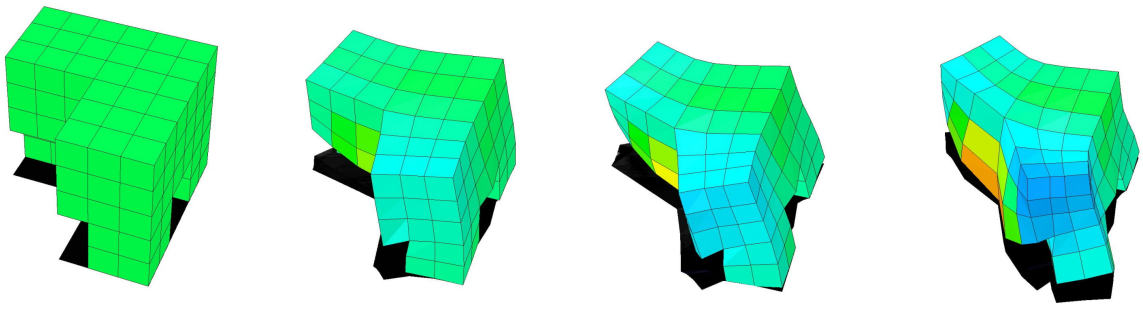


Figure 6.7: This damaged robot (amp. = 1/4 body) contracted its hips and expanded its pelvis to recover function (**youtu.be/UBvsR6tZf5c**).

In the case where half of the robot's body is removed, the undeformed structure falls under gravity onto its side; one local optima was thus to crawl "facedown". A better strategy was found in which the robot could remain upright by using the two remaining limbs as forelegs and expanding the stump to form a wide hind leg. An equally proficient strategy was observed in which the robot diminished one or both of its legs, expanded its spine, and moved longitudinally (Fig. 6.8).

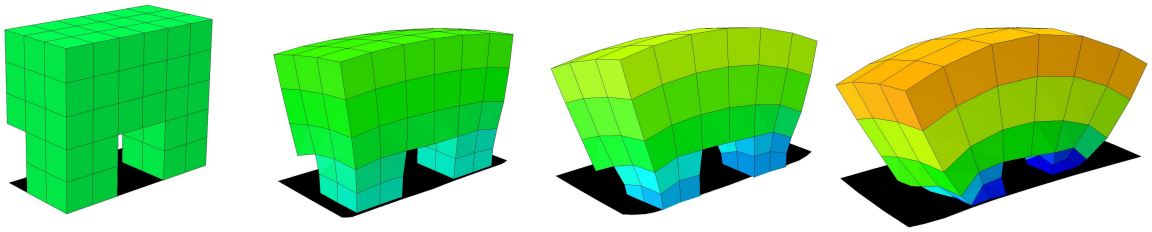


Figure 6.8: Shape change in this damage case (amp. = 1/2 body) enabled upright, lengthwise movement, instead of falling over (**youtu.be/nfCaVZVBmKI**).

There were many successful variations on this theme, but one of the best designs in this case did the exact opposite: The robot expanded its remaining limbs to their maximum volume, contracted its spine, and flipped over (once) to walk longitudinally with the added momentum generated from large, swaying front and back limbs (Fig. 6.9).

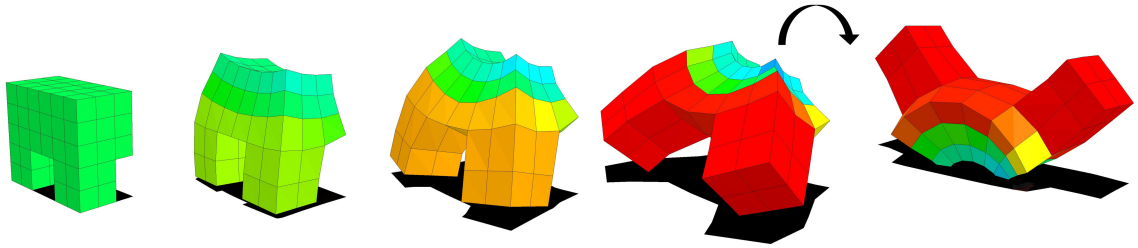


Figure 6.9: This damaged robot (amp. = 1/2 body) contracted its spine, expanded its limbs, and flipped over onto its back to walk lengthwise and exploit the elastic properties of its new arms (**youtu.be/WwYdSnuJBBA**).

However, after the most extreme insult, when all but a quarter of the robot is lost, there is insufficient material to regenerate legs or execute other more extreme shape changes. Neither recovery option cultivated (visually) appreciable gains in fitness. Yet, while this case removes 71% of the original volume it is significantly *less* deleterious to controller optimization than amputating the four legs, which removes just 23% of the original volume. Insult is thus a matter of kind, not degree.

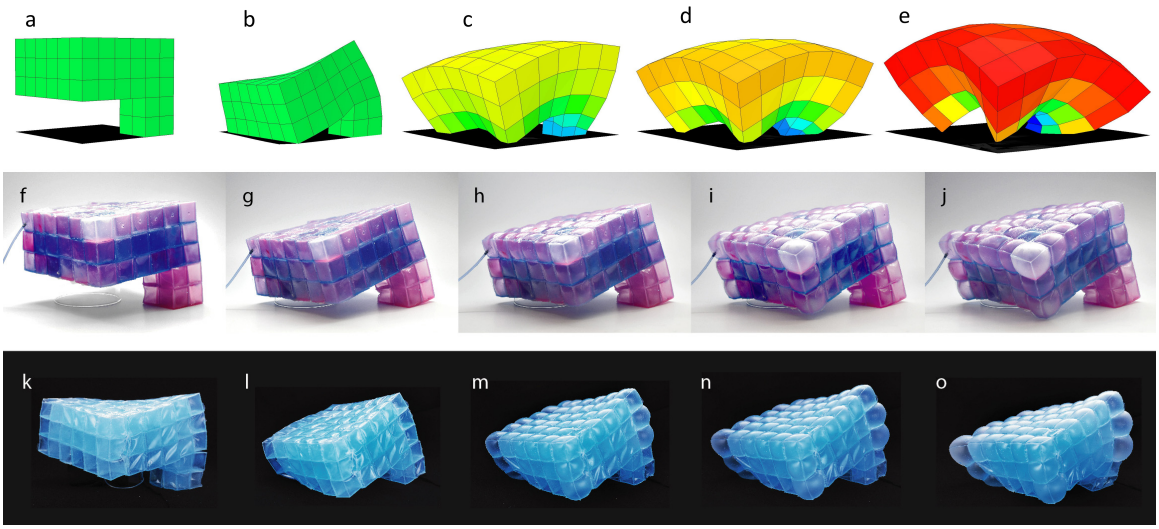


Figure 6.10: After losing three legs to injury (amp. = 3 legs), the former quadruped is reduced to a monopedal structure (**a**), the shape of which was then optimized for locomotion speed, resulting in an expanded spine, the folding-inward of the remnant predamage leg, and the "regeneration" of the three missing legs (**c**-**e**). This simulated strategy was then realized in two implementations using pneumatically-actuated, cubic elastomer bladders. The purple robot (**f**-**j**) consists of two layers of drip-molded silicone; the blue robot (**k**-**o**) consists of a single layer, and is thus less stable but more deformable. A single air inlet here yields the rudiments of appropriate shape change, but pressure oscillations in this setup did not yield locomotion (**youtu.be/A2KTGhCFxK8**).

The transferal of recovery strategies to reality.

To investigate the potential for directly transferring recovery strategies from simulation to reality, we aimed to transfer the overall shapes that are pictured in Figs. 6.1 and 6.10. In these particular cases, the optimizer found shapes with contiguous sections of voxels actuated to similar levels. Thus, we here examine the one-actuator case, in which the voxels with the largest rest volumes—the top layer of $(6 \times 6 = 36)$ voxels and the two corner voxels just below the top layer, on each corner of the torso (8 in total)—were connected to the same air inlet. Voxels not hooked into the air line were punctured to allow for passive deflation and contraction of the robot, mimicking the simulated robot's ability to contract voxels by decreasing the rest lengths b_i.

The purple robot adequately expands the top layer of voxels in both cases (Figs. 6.1j and 6.10j), but fails to reach the overall target shapes drawn in Figs. 6.1e and 6.10e. Although further increasing pressure did indeed lead to larger deformations, the outer voxels inflate farther than interior ones, limiting the maximum viable actuation pressure. The thinner voxel walls of the blue robot exacerbated this issue (Fig. 6.10k-o), but their increased flexibility enabled a more faithful transferal of overall surface

curvature. Another limitation we discovered was friction. Fully realizing the target shape in Fig. 6.10e requires the robot to drag its leg inward across the floor, tucking it under its body; but the silicone leg often stuck to the surface, preventing the prescribed maneuver in reality.

The silicone design and 1-axis rotational molding technique are still quite promising. Even when inflated at high enough pressures to make the outer voxels approximately spherical (Fig. 6.10o), the voxels did not rupture. To achieve more consistent expansion of interior and exterior voxels, the later should be inflated at a lower pressure than the former. By incorporating strain sensors [250] and closed-loop control in future, the robot could correct for this variation on the fly. By actuating different voxels at different pressures, and enabling active contraction in addition to expansion, a much wider range of simulated shapes could be attained in reality.

6.4 Discussion

In this paper, a new approach to robot damage recovery has been proposed. Instead of presenting the remnant shape of the damaged robot to optimization as fixed, we enable optimization to change this shape as the essential part of the recovery process. In doing so we realized a machine that recovered more function than an otherwise equivalent system that could adapt its controller but not deform its shape.

In future work we will improve the transferal of simulated morphing machines to physical ones using existing sim2real methods [21, 27, 55, 104, 127, 230] adapted appropriately to meet the additional transferal demands dictated by soft materials [149]. We will also generalize our optimization method such that control and shape readaptation can be combined as dictated by the form of damage, predamage structure of the robot, and its task environment.

Biological regeneration.

In past work, rigid-bodied robots have been venerated for their ability to "adapt like animals" [27, 55]. These machines, which were constructed from undeformable metals and hard plastics, automatically learned to control their bodies in spite of missing or broken legs. But when an animal loses one or more of its legs to injury, it does not adapt by merely searching for a new mental representation of behavior that successfully maps onto the damaged body. Rather, they often fundamentally deform their damaged "hardware" into something more controllable.

Evidence for this abounds. A famous example is the congenitally two-legged goat described by Slijper [217]: an otherwise normal goat which was born without forelegs adopted an upright posture and learned to walk on its hind legs alone. In addition

to enlarged hind legs, striking changes in morphology were documented, including a greatly elongated gluteal tongue and an innovative arrangement of small tendons, a narrowed pelvis, an oval (rather than V-shaped) thoracic cross-sectional shape, a curved spine, and an unusually large neck [249]. The animal's body resembled that of a kangaroo more closely than that of a normal goat.

Other animals can regenerate. The planarian flatworm can be cut into many pieces (the record is 279) all of which grow back to a full organism, regenerating not just tail and head, but eyes and the complete nervous system [159]. Vertebrates, such as frogs, also display the capability of regenerating limbs, jaws, eyes and a variety of internal structures [32]. Humans too (especially children) are sometimes capable of fingertip regeneration after distal phalange amputation [105].

Mechanisms of biological regeneration.

Several of the mechanisms by which organisms achieve these forms of self-editing of their own anatomy pose design challenges and future research directions for robotics.

First is the ability to harness the behavior of low-level components (cells) towards a specific large-scale goal-state: salamanders can regenerate whole limbs, eyes, tails, ovaries, and other organs [151], but growth and remodeling ceases when a correctly shaped and sized organ is complete [186]. Second is the flexibility and robustness of systems under novel conditions. For example, tadpoles whose facial organs are experimentally placed in abnormal configurations will undergo novel rearrangements to still give rise to normal frog faces during metamorphosis [239], showing that the genome encodes not a hardwired set of movements for each organ but rather specifies a machine that can remodel toward the same target morphology from a variety of unexpected starting states. Thus, it is critical to understand and exploit the ability of evolution to give rise to hardware that is well-adapted to the normal environment but also retains significant plasticity [225].

Third is the fact that during regeneration, the tissues making growth and morphogenesis decisions are themselves being drastically rearranged: thus, the computational control circuitry *is itself* the object of the deformation actuators, forming a closed loop in which information is reliably processed in a medium that is constantly changing [185]. Finally, the remarkable robustness of morphological computation extends to information learned within the lifetime of the organism [25]. Butterflies, which result from a caterpillar brain that is almost completely dissolved during metamorphosis, still remember information learned during the caterpillar stage [24]. Flatworms, which regrow their entire heads, still remember information they learned prior to decapitation [53, 212].

Attempts to implement these capabilities in artificial systems (whether robotic or via synthetic biology) are likely to enrich not only engineering technology, but also

to feed back to the biological sciences and biomedicine. The current understanding of computation in biological tissues has numerous gaps, which are only likely to be filled by attempts to build these capabilities from the ground up [110].

Metamorphosing machines.

It has been shown here that robots, too, are not only capable of regenerating limbs, but that such deformation can manifest by selecting for function recovery alone, instead of a target legged shape.

However, this ability largely depends on the material with which robots are made, for even if morphology is free to change in rigid bodies, the ways in which such change can occur are limited at best. In [28], robots used a combination of rotary and linear actuators to slowly angle appendages downward and extrude them outward, thus simulating limb growth. In softer machines, there are more ways for morphology to change: The soft robot used here was able to locally deform its geometry to bend, twist, compress or expand throughout its body. Its also possible, although not investigated here, for soft robots to change their material properties, such as stiffness, through (e.g.) granular jamming [34, 123].

The possibility of this latter change highlights the inadequacy of the name "soft robot". When a granular jamming robot jams (removes excess internal air to become stiff) does it cease to be a soft robot? What if it never unjams? For the purposes of damage repair, the most important property of soft robots is not that they are soft *per se*, but that they may easily change their structural and material properties (possibly including stiffness). One can envisage future "rigid" nanobots capable of self-assembling into a protean metamachine that can rearrange so as to regrow a lost part; but that day seems far off, whereas soft robots, capable of continuous morphological change, are already becoming a reality.

The future of this line of work promises not just new robotic systems but also new science. Shapeshifting robots, recast as scientific tools, can shed new light on old biological questions about developmental plasticity, regeneration and homeostasis [121, 122, 142]. And, symmetrically, new theories about the mechanisms that lie at the heart of such questions can be physically instantiated and optimized in a new breed of useful, autonomous and adaptive machines.

Chapter 7

Living Protean Machines

Appeared as:
S. Kriegman et al., A scalable pipeline for designing reconfigurable organisms. *Proceedings of the National Academy of Sciences (PNAS)* **117** (4) 1853–1859 (2020).

Significance:
Most technologies are made from steel, concrete, chemicals and plastics, which degrade over time and can produce harmful ecological and health side effects. It would thus be useful to build technologies using self-renewing and biocompatible materials, of which the ideal candidates are living systems themselves. Thus, we here present a method that designs completely biological machines from the ground up: computers automatically design new machines in simulation, and the best designs are then built by combining together different biological tissues. This suggests others may use this approach to design a variety of living machines to safely deliver drugs inside the human body, help with environmental remediation, or further broaden our understanding of the diverse forms and functions life may adopt.

Abstract:
Living systems are more robust, diverse, complex, and supportive of human life than any technology yet created. However, our ability to create novel lifeforms is currently limited to varying existing organisms or bioengineering organoids in vitro. Here we show for the first time a scalable pipeline for creating functional novel lifeforms: AI methods automatically design diverse candidate lifeforms in silico to perform some desired function, and transferable designs are then created using a cell-based construction toolkit to realize living systems with the predicted behaviors. Although some steps in this pipeline still require manual intervention, complete automation in future would pave the way to designing and deploying unique, bespoke living systems for a wide range of functions.

7.1 Introduction

Most modern technologies are constructed from synthetic rather than living materials because the former have proved easier to design, manufacture, and maintain; living systems exhibit robustness of structure and function and thus tend to resist adopting the new behaviors imposed on them. However, if living systems could be continuously and rapidly designed ab initio and deployed to serve novel functions, their innate ability to resist entropy may enable them to far surpass the useful lifetimes of our strongest yet static technologies. As examples of this resistance, embryonic development and regeneration reveal remarkable plasticity, enabling cells or whole organ systems to self-organize adaptive functionality despite drastic deformation [23, 239]. Exploiting the computational capacity of cells to function in novel configurations suggests the possibility of creating synthetic morphology that achieves complex novel anatomies via the benefits of both emergence and guided self-assembly [110].

Currently, there are several methods underway to design and build bespoke living systems. Single-cell organisms have been modified by refactored genomes, but such methods are not yet scalable to rational control of multicellular shape or behavior [103]. Synthetic organoids can be made by exposing cells to specific culture conditions but very limited control is available over their structure (and thus function) because the outcome is largely emergent and not under the experimenter's control [203]. Conversely, bioengineering efforts with 3D scaffolds provide improved control [169, 176, 231], but the inability to predict behavioral impacts of arbitrary biological construction has restricted assembly to biological machines that resemble existing organisms, rather than discovering novel forms through automatic design.

Meanwhile, advances in computational search and 3D printing have yielded scalable methods for designing and training machines in silico [46, 214] and then manufacturing physical instances of them [27, 38, 140]. Most of these approaches employ an evolutionary search method [164] that, unlike learning methods, enables the design of the machine's physical structure along with its behavior. These evolutionary design methods continually generate diverse solutions to a given problem, which proves useful as some designs can be instantiated physically better than others. Moreover, they are agnostic to the kind of artefact being designed and the function it should provide: the same evolutionary algorithm can be reconfigured to design drugs [63], autonomous machines [38, 140], metamaterials [101], or architecture [163].

Here, we demonstrate for the first time a scalable approach for designing novel living systems in silico using an evolutionary algorithm, and we show how the evolved designs can be rapidly manufactured using a cell-based construction toolkit. The approach is organized as a linear pipeline that takes as input a description of the biological building blocks to be used and the desired behavior the manufactured system should exhibit (Fig. 7.1). The pipeline continuously outputs performant living

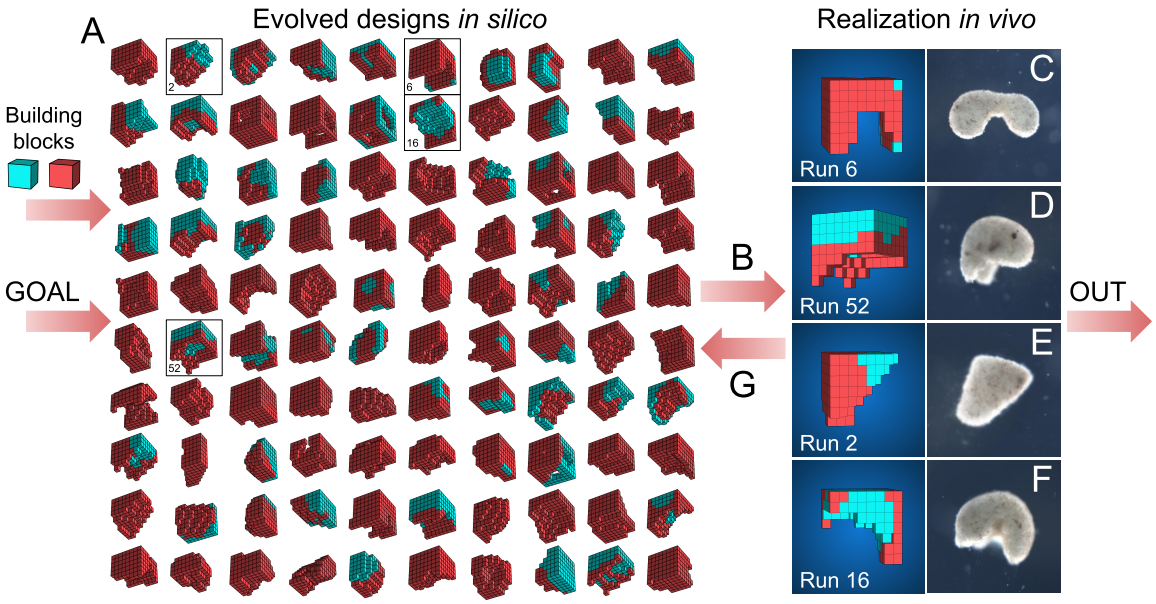


Figure 7.1: **Designing and manufacturing reconfigurable organisms.** A behavioral goal (e.g., maximize displacement), along with structural building blocks (here, contractile (red) and passive (cyan) voxels), are supplied to an evolutionary algorithm. The algorithm evolves an initially random population and returns the best design that was found. The algorithm is rerun 99 times starting with different random populations, generating a diversity of performant designs *in silico* (**A**). Performant designs are then filtered by their robustness to random phase-modulation of their contractile cells (**B**), constructed *in vivo* using developing *Xenopus* cardiomyocyte and epidermal cell progenitors (**C**-**F**), and placed on the surface of a petri dish where their behavior is observed and compared to the design's predicted behavior. Discrepancies between *in silico* and *in vivo* behavior are returned to the evolutionary algorithm in the form of constraints on the kinds of designs that can evolve during subsequent design-manufacture cycles (**G**). Concurrently, tissue layering and shaping techniques are modified such that realized living systems behave more like their virtual model.

systems that embody that behavior in different ways. The resulting living systems are novel aggregates of cells that yield novel functions: above the cellular level, they bear little resemblance to existing organs or organisms.

7.2 Results

The pipeline is organized as a sequence of generators and filters (Fig. S1). The first generator is an evolutionary algorithm that discovers different ways of combining the biological building blocks together to realize the desired behavior. A population of random designs are first created. Then, each design is simulated in a physics-based virtual environment and automatically assigned a performance score. Less performant designs are deleted and overwritten by randomly-modified copies of more performant

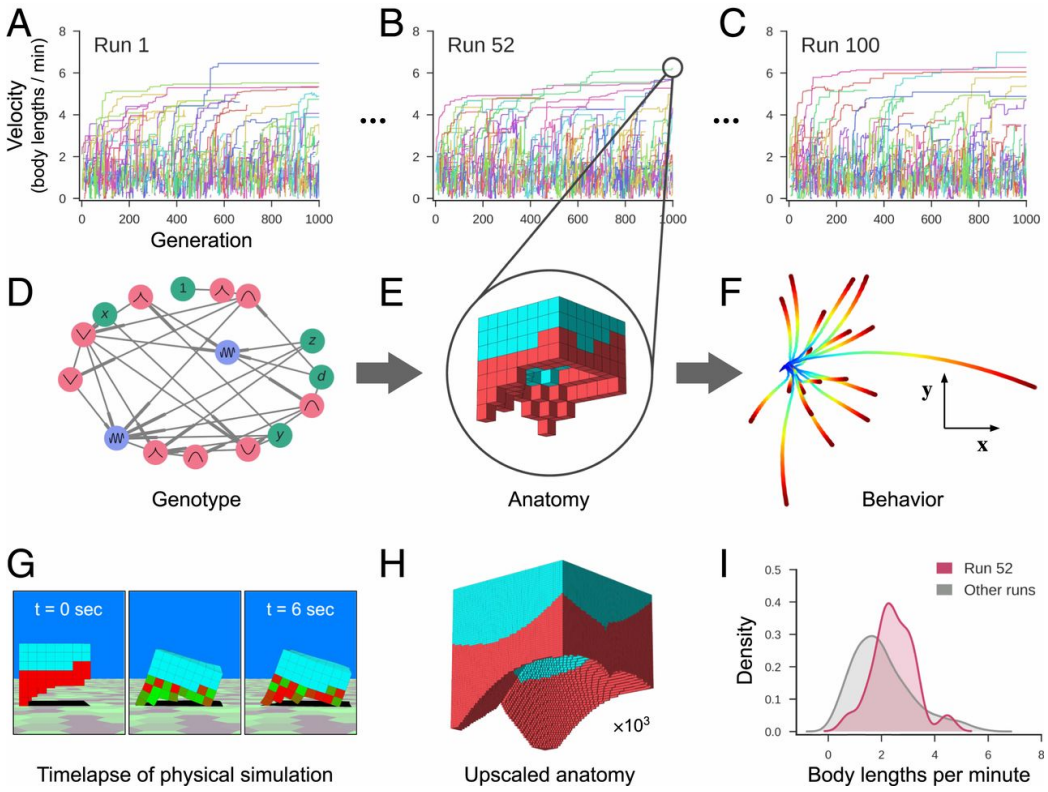


Figure 7.2: **Designing reconfigurable organisms.** For a given goal, 100 independent evolutionary trials were conducted in silico (**A-C**). Each colored line represents the velocity of the fastest-moving design within its clade. Each genome (**D**) dictates anatomy and behavior by determining where and how voxels are combined, and whether they are passive (cyan) or contractile (red; **E**). Genomes simulate a developmental process and are described in more detail in Sect. S4. The differing behavioral traces produced by a design (**F**) are a result of randomly perturbing the actuation of each contractile cell during each evaluation period. The behavioral traces all originate from the same position (blue) but diverge over time until their final destination (red). **G:** During one evaluation period, after settling under gravity for 1 sec, compressed and expanded contractile voxels are shown in red and green respectively. Because the genotype is scale-free, the anatomical resolution of any design can be increased (**H**) while preserving geometry (but not necessarily behavior). When all evolutionary trials complete, the most performant design from each trial is extracted (**I**). The robust design passed to the next stage of the pipeline moves, on average, more rapidly (red curve) than the average speed of the other 99 designs (gray curve).

designs. Repeating this process yields populations of performant and diverse designs (Fig. 7.2).

As there are likely to be many differences between the simulated and targeted physical environments, performant designs are passed through a robustness filter which only allows passage of designs that sustain the desired behavior in the face of noise (Sect. S7). Previous work has shown that noise resistance in simulation is a simple and effective predictor of whether a design will maintain its behavior when

instantiated physically [107]. The surviving noise-resistant designs are then passed through a build filter (Fig. S4) which removes designs that are not suitable for the current build method (Fig. S6) or unlikely to scale to more complex tasks in future deployments. The manufacturability of a design depends on the minimal concavity size that will persist in aggregations of developing stem cells, which tend to close small gaps in their collective geometry (Fig. S7). The scalability of a design depends on its proportion of passive tissue, which provides space for future organ systems or payloads (Fig. S13).

The designs that successfully pass through the build filter are then built out of living tissues. Pluripotent stem cells are first harvested from blastula stage Xenopus laevis embryo, dissociated, and pooled to achieve the desired number of cells. Following an incubation period, the aggregated tissue is then manually shaped by subtraction using a combination of microsurgery forceps and a 13 micron wire tip cautery electrode, producing a biological approximation of the simulated design. Further, contractile tissue can be layered into the organism through the harvesting and embedding of Xenopus cardiac progenitor cells, an embryonically derived cell type which naturally develops into cardiomyocytes (heart muscle) and produces contractile waves at specific locations in the resultant shaped form (Fig. S6).

The final product of this procedure is a living, three-dimensional approximation of the evolved design, which possesses the ability to self-locomote and explore an aqueous environment for a period of days or weeks without additional nutrients. These organisms are then deployed into their physical environment, and resultant behavior, if any, is observed (Fig. 7.3). Behaviors are then compared against those predicted by their simulated counterparts to determine whether or how well behaviors transferred from silico to vivo (Fig. 7.4).

After several organisms have been deployed and observed, it is likely that they exhibit varying amounts of the desired behavior. Common patterns among the successful systems are distilled down into constraints and supplied back to the evolutionary algorithm, which now evolves designs that are not just performant but also conform to the constraints (Sect. S6). This increases the success likelihood of subsequent design-to-deployment attempts. Reconfigurable organisms were evolved to exhibit four different behaviors: locomotion, object manipulation, object transport, and collective behavior (Sect. S10). To achieve this, the pipeline was employed four times.

Locomotion.

To obtain a diverse population of designs, 100 independent trials of the evolutionary algorithm were conducted (Fig. 7.2A-C), each starting from a different set of initial random designs. During each trial, designs were selected based on net displacement

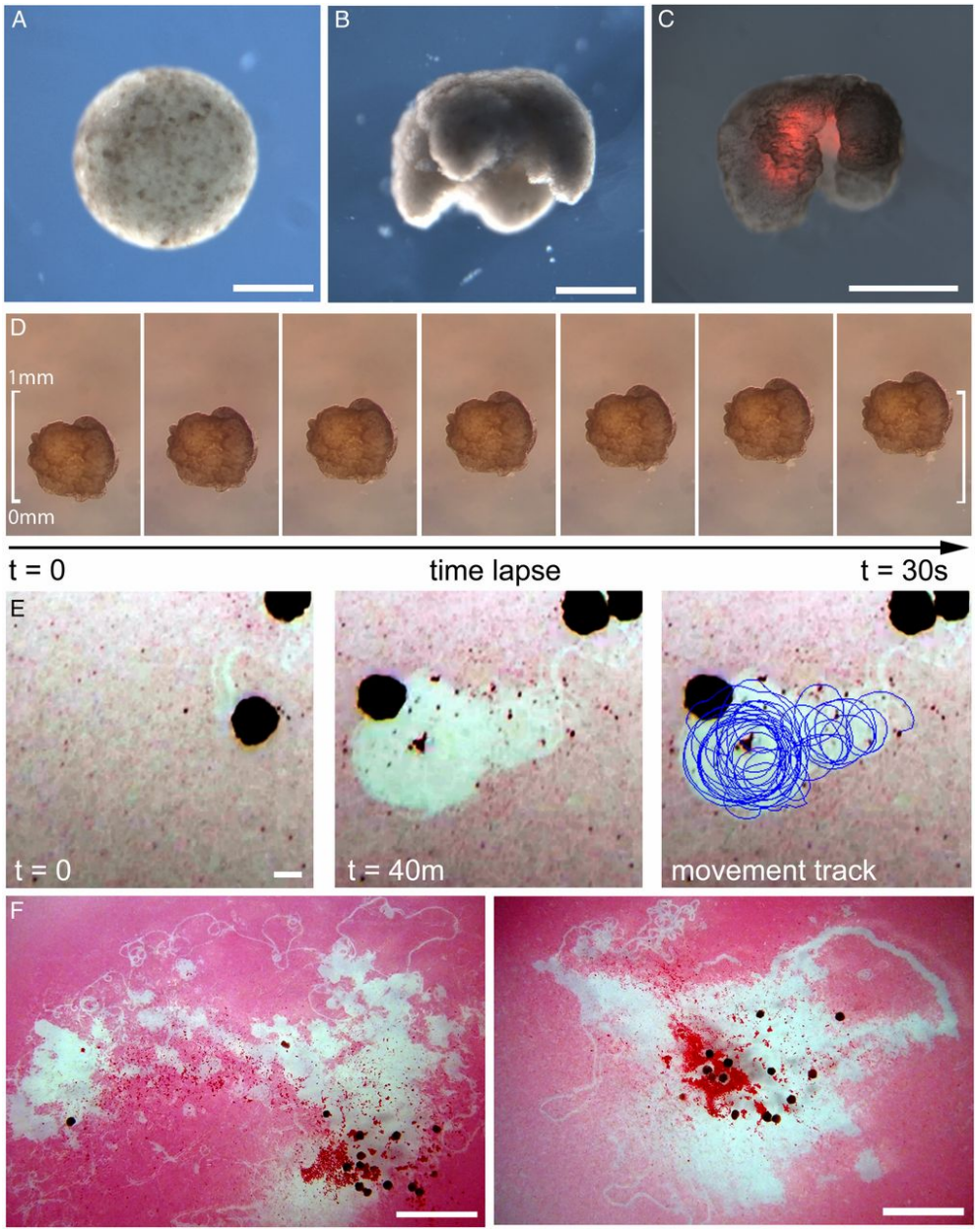


Figure 7.3: **Manufacturing reconfigurable organisms.** (**A**) Aggregation of pluripotent blastula cells harvested from Xenopus laevis embryos. (**B**) Shaping results in three-dimensional representations of the evolved in silico designs. (**C**) Layering of cardiac progenitor cells results in contractile cardiomyocyte tissue at specific locations, visualized by red fluorescent lineage tracer. (**D**) Time-lapse imaging of self-locomotion in an aqueous environment. (**E**) Emergent behavior of debris aggregation by an individual within the environment and (**F**) by groups of biological representations over a 24h period (Sect. S10.4). Scale bars indicate 500 μm for A-E and 5 mm for F, respectively.

achieved during a 10 second period (with randomized, phase-modulated contraction, cycling at 2 Hz). Additional selection pressures were applied to maintain diversity by inducing competition within and between unique genetic lineages within each trial [204], yielding unique ecological dynamics (Sect. S5). The most fit designs at the end of each trial were extracted (Fig. 7.1A) and passed through the robustness and build filters (Fig. S4). During this filtering process, buildable and scalable designs that retain rapid locomotion during random perturbations are selected for manufacture (Figs. 7.3 and S6).

Cilia, which produce locomotion through metachronal waves (the generation of sequential and directional propagating waves, as opposed to synchronized beating), were not modeled in silico and were suppressed in vivo through embryonic Notch ICD mRNA microinjection [58]. Thus, any displacement results from contractile cardiac muscle tissue that pushes against the surface of the dish. This simplifies the simulation and its comparison to the realized organism. Trajectories of deciliated designs are compared in silico and in vivo, in two orientations (upright and inverted 180° about the transverse plane) thus isolating the impact of the designed morphology on the difference between predicted and realized behavior. For at least one design, the data suggest that the desired behavior successfully transferred when it was upright but not when inverted (Fig. 7.4). More specifically, the upright organisms' direction of movement matched that of the in silico design under random perturbations ($p<0.01$; details in Sect. S9), and inverting the design significantly reduced its net displacement both in silico ($p<0.001$) and in vivo ($p<0.0001$). This suggests that successful transference did not result by chance but rather due to the design itself.

Object manipulation.

When the environment is strewn with particulate matter, motile designs spontaneously aggregated the external objects both in silico (Fig. S10) and in vivo (Figs. 7.3F and S11). More precise object manipulation can be selected for as an explicit goal, such as specifying target areas from which debris should be cleared, or target objects to discard. The latter goal was implemented and primitive end-effectors evolved in simulation (Fig. S12).

Object transport.

Some designs evolved for displacement reduced hydrodynamic drag (see Sect. S6) via a hole through the center of their transverse plane. This more complex topology was realized in vivo (Fig. S13) but was not layered with contractile tissue. In simulation, this emergent feature can be exapted as a pouch to store and transport objects. In a subsequent round of evolution, pouches were explicitly incorporated as a design

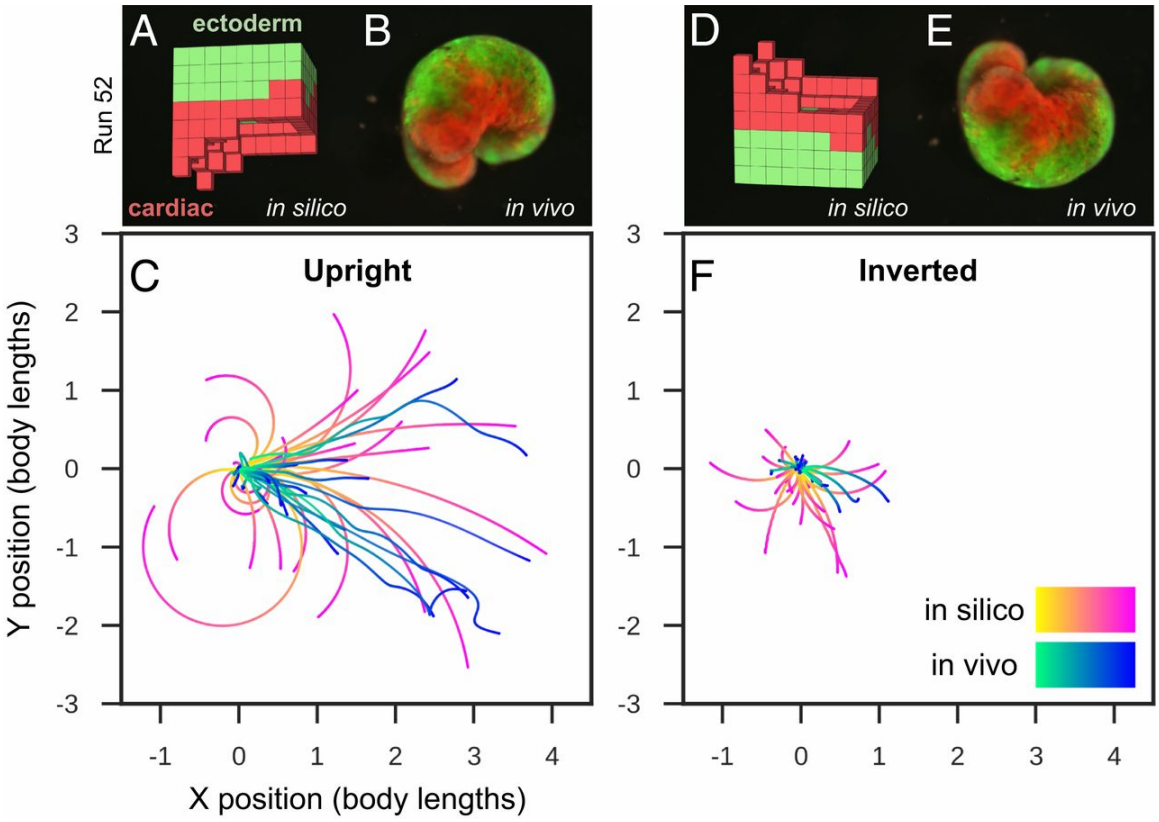


Figure 7.4: **Transferal from silico to vivo.** The first design selected for fabrication and specific hypothesis testing (**A**) was the most robust yet stable and energy-efficient configuration of passive (epidermis; green) and contractile (cardiac; red) tissues found by the evolutionary algorithm. The design was evaluated 25 times for 1 minute of simulation time, resulting in 25 movement trajectories (pink curves in **C**). Six reconfigurable organisms were built which embodied this design (e.g., **B**) (Sect. S9). Three were evaluated four times and the other three were evaluated five times for 10 minutes each (27 blue curves in C). The organisms' direction of movement matched the design's predicted direction of movement ($p<0.01$; details in Sect. S9). To determine whether the organisms' movement was a result of chance or due to the design's evolved geometry and tissue placement, geometry and tissue distribution was altered by rotating the design 180° about its transverse plane (**D**) and evaluating it another 25 times in silico (pink curves in **F**). Each of the six organisms were likewise inverted (**E**): four were evaluated five times while the remaining two were only evaluated once (22 blue curves in F). Inverting the design significantly reduces its net displacement ($p<0.001$), as did inverting the organisms ($p<0.0001$).

constraint, and the new goal of maximizing the distance of the carried object was employed. This yielded evolved object transport in silico (Fig. S13).

Collective behavior.

Multiple designs can be placed in the same environment, yielding collective behavior [248] (Figs. S10 and S11). Several such behaviors predicted in silico were observed in vivo. For instance, two designs often collide, form a temporary mechanical bond, and orbit about each other for several revolutions before detaching along tangential trajectories (Fig. S10). This phenomenon is more pronounced when cilia are not inhibited on the organisms: individuals frequently become entangled with their neighbors, often changing partners across an observation (Figs. 7.3F and S11).

7.3 Discussion

Although simulation and design of rigid structures and machines has been possible for some time, only recently has it become computationally tractable to simulate the combined behavior of arbitrary aggregates of soft components with differing material and actuation properties [95]. As shown for the first time here, such fine-grained simulations can be embedded in evolutionary search methods to discover designs that can be instantiated in biological, rather than artificial materials.

The resulting organisms embodied not only the structure (Fig. S8) of evolved in silico designs but also their behavior (Fig. 7.4), despite modeling cardiomyocyte temporal coordination as random noise. As a side effect of selection pressure for locomotion, derandomizing morphologies evolved: evolutionary improvement occurred through changes in overall shape, and distribution of the passive and contractile cells, to collectively derandomize the global movement produced by the random actuation. In biology, such robustness to random noise is ubiquitous; one example is the ability of many species to adapt to wide ranges of diversity in cell size and number as starting points in their embryogenesis [52].

The behavioral competence of individual cells, and the propensity of cells to cooperate in groups, facilitate functional morphogenesis in novel circumstances. The lifeforms presented here, despite lacking nervous systems, following novel developmental trajectories, and being composed of materials from different tissues, nevertheless possess these self-organizing properties. These properties synergize with and support the behavior they were designed to exhibit. For instance, although signaling between cardiomyocytes was not enforced, emergent spontaneous coordination among the cardiac muscle cells produced coherent, phase-matched contractions which aided locomotion in the physically-realized designs. Also, some of the designs, when combined, spontaneously and collectively aggregate detritus littered within their shared environment (Figs. 7.3F and S11). Finally, reconfigurable organisms not only self-maintain their externally-imposed configuration, but they also self-repair in the face of damage, such as automatically closing lacerations (Fig. S9). Such spontaneous

behavior cannot be expected from machines built with artificial materials unless that behavior was explicitly selected for during the design process [124].

This approach admits future generalization and automation because the generator-and-filter architecture enables modular addition, removal, or reorganization of elements in the pipeline for rapid design and deployment of new living systems for new tasks in new domains. For instance, a filter could be added which pre-emptively steers the evolutionary algorithm away from portions of the design space known to contain designs that cannot be realized physically [117]. Or, inspired by the hierarchical organization of deep neural networks [254], individual designs output by one generator could become the building blocks input to the next generator, thus enabling hierarchical design and re-use of cellular assemblies, and assemblies of assemblies.

Beyond the applications reported here, the generality of this approach is as of yet unknown. But, advances in machine learning, soft body simulation, and bioprinting are likely to broaden the potential applications to which it may be put in future. Applications could be numerous, given the ease of misexpressing novel proteins and synthetic biology pathways and computational circuits in Xenopus cells [237]. Given their non-toxicity and self-limiting lifespan, they could serve as a novel vehicle for intelligent drug delivery [181] or internal surgery [135]. If equipped to express signaling circuits and proteins for enzymatic, sensory (receptor), and mechanical deformation functions, they could seek out and digest toxic or waste products, or identify molecules of interest in environments physically inaccessible to robots. If equipped with reproductive systems (by exploiting endogenous regenerative mechanisms such as occurs in planarian fissioning), they may be capable of doing so at scale. In biomedical settings, one could envision such biobots (made from the patient's own cells) removing plaque from artery walls, identifying cancer, or settling down to differentiate or control events in locations of disease. A beneficial safety feature of such constructions is that in the absence of specific metabolic engineering, they have a naturally limited life-span.

These methods, reagents, and data extend the breadth of model organisms available for study by designing living systems that are as orthogonal as possible to existing species, yet capable of being built from existing cell types. By enabling a computationally-guided interplay between emergent and designed processes, this platform facilitates studies of the relationship between genomes (in our case, wild-type Xenopus laevis), the resulting body-plan, and its behaviors in diverse environments. Thus, such reconfigurable organisms could serve as a unique model system facilitating work in the evolution of multicellularity, exobiology, artificial life, basal cognition, and regenerative medicine. If equipped with electrically-active cells and selected for cognitive or computational functions [13], such designed systems may similarly broaden our understanding of how intelligence can be instantiated in living as well as non-living systems.

7.4 Materials and Methods

Evolutionary Design.

Designs (Sect. S2) were evolved inside a physics engine (Sect. S3) as reconfigurable aggregations of passive and contractile voxels (Fig. 7.1). On the first pass through the pipeline using the goal behavior of locomotion, we simulated designs on land and allowed the evolutionary process to finely tune their actuation. This resulted in highly-performant but non-transferable designs (Fig. S2) with powerful, bounding gaits that are not obtainable in vivo with the current build method (Sect. S8). These gaits were characterized by timeframes (on average, 47% of the gait cycle) in which no part of the in silico design was in contact with the simulated ground plane. In vivo, however, the deciliated organisms always kept part of their ventral surfaces in contact with the surface of the dish due to negative buoyancy.

These discrepancies were rectified by adding constraints into the pipeline in the form of adjustments to environmental and actuation settings, which were altered as follows. On the second pass, the fidelity of the simulated environment was increased by incorporating first-order hydrodynamics: the modified environment consisted of an infinite plane submerged in water, which was approximated by decreasing the coefficient of gravitational acceleration (increasing buoyancy) and applying a drag force to each voxel face on the design's surface (Sect. S6). Secondly, actuation was randomized: contractile cells were revised to have random phase-offsets from a central pattern generator (a sine wave with frequency 2 Hz). More specifically, each voxel of a randomly-configured design (one of which was injected into the population at each generation; Sect. S5) was assigned a random phase offset, which was held fixed in its descendants (the entire clade). Mutations switched each voxel to be either present or absent, and, if present, either passive or active (contractile), but the original phase offset, at every location in the workspace, was hardcoded. This reduced the dependence on precisely-timed excitation, and promoted the discovery of more robust mechanical structures (Fig. S3).

The behavior of designs generated on the second pass better matched the behavior of the actual living systems: on average, designs were in contact with the ground plane for 93.3% of their evaluation period, compared to just 52.7% on the first pass (Sect. S6).

Robustness Filter.

The most performant designs (Fig. 7.1A) were sorted by their robustness to random perturbations in their actuation. Phase offsets stored in the genotype were mutated by adding a number that was drawn randomly from a normal distribution with mean

zero and standard deviation $s = 0.4\pi$ (which is 40% of the $-\pi/2$ to $\pi/2$ range of valid phase-offset values). This hyperparameter was selected to be large enough to scramble the original phase-offset value without being so large as to push all mutations up against the $\pm\pi/2$ bounds. Designs that maintained the highest average performance across this actuation noise were passed, one by one, in order of their robustness ranking, to the build filter.

Build Filter.

The most robust designs are evaluated by their manufacturability under the current build method, which layers contiguous tissue regions sequentially (Fig. S6). The minimal concavity was examined by producing organisms with progressively smaller shape deformations, then determining which persist across the lifespan of the organism, and which close due to tissue contraction, leading to loss of concavity. Preliminary work determined that concavities with a width of 100μm or greater (12% of total body length) produced stable long term deformations suitable for biological building (Fig. S7).

Additionally, the build filter removes designs that are more than 50% muscle, in order to reserve sufficient design space to add specialized cells for purposes other than locomotion, including sensory input, metabolism, memory, biosensors, etc. Also, contractile tissue incurs a much higher metabolic cost compared to non-muscle tissue (the human heart consumes approximately 1mM ATP per second; [190]). Thus, limiting this tissue type increases the total lifetime of transferred designs. The most robust designs that satisfy these selection criteria (Fig. S4) are passed through the build filter to the next stage of the pipeline: the realizability generator.

Realizability Generator.

Reconfigurable organisms were created using Xenopus embryos as donor tissue under methods approved by the Institutional Animal Care and Use Committee and Tufts University Department of Laboratory Animal Medicine under protocol number M2017-53.

Fertilized Xenopus laevis eggs were reared in a 0.1X, p.H. 7.8, Marc's Modified Ringers solution (MMR) using standard protocols and staged according to Neiuwkoop and Faber [171, 216]. For shaping experiments, animal caps were manually cut at St. 9 using surgery forceps (Dumont, 11241-30 #4) and transferred to Calcium and Magnesium free medium for five minutes (50.3 mM NaCl, 0.7 mM KCl, 9.2 mM Na2HPO4, 0.9 mM KH2PO4, 2.4 mM NaHCO3, 1.0 mM EDTA, pH 7.3). The outer ectoderm layer was manually removed and discarded, while the inner layer was agitated until fully dissociated (cells are this stage are largely pluripotent, but

differentiate into ectoderm without further intervention). Material from five animal caps was pooled and transferred to a welled dish containing 0.75x MMR. After 24H at 14°C, the spherical re-aggregate was moved to a clean 1% agarose coated dish containing 10ml 0.75x MMR and 5μl gentamicin (ThermoFisher Scientific, 15710072). Forty eight hours after tissue re-aggregation the resulting tissue (now fated to become specific epidermal cell lineages including ionocytes, small secretory cells, and goblet cells), was shaped using a combination of microsurgery forceps and a MC-2010 micro cautery instrument with 13 micron wire electrodes (Protech International Inc., MC-2010, 13-Y1 wire tip cautery electrode). Tissue was reshaped as necessary for three hours to create the desired anatomical outcome, after which it was moved to a clean 1% agarose coated dish containing 10ml 0.75x MMR and 5μl gentamicin and raised at 14°C.

For contractile movement experiments, cohorts of Xenopus embryos were microinjected with one of two synthetic mRNAs at the four cell stage using standard protocols [171]. mRNA for the fluorescent lineage tracer tdTomato [246] and the multiciliated cell inhibitor Notch ICD [16, 58] was synthesized using mMESSAGE transcription kits (TheromoFisher Scientific, AM1340). Injections were performed in 3% Ficoll solution using a pulled capillary to deliver 370pg of mRNA for each transcript to all four cells. tdTomato microinjected embryos were reared for at 22°C while Notch ICD injected embryos were reared at 14°C. Twenty four hours after injection, stage 10 Notch ICD injected embryos were moved to a 1% agarose coated petri dish containing 0.75x MMR, and animal caps were manually cut using surgery forceps as above. In addition, stage 23-24 tdTomato injected embryos were transferred to the same dish and the presumptive heart-field was excised with the outer layer of ectoderm then removed and discarded. Presumptive heart tissue was then placed between two Notch ICD injected animal caps, and the three layers were allowed to heal for one hour at 22°C. Following healing, the tissue was moved to clean 1% agarose coated dish containing 10ml 0.75x MMR and 5μl gentamicin and raised at 14°C. For shaping, resultant tissue was sculpted as above using a combination of microsurgery forceps and a MC-2010 micro cautery instrument.

Transferability Filter.

All samples were imaged live in 0.75x MMR at 20°C using a Nikon SMZ-1500 microscope equipped with both top and substage illumination. Still Images were captured on a QImaging Retiga 2000R CCD camera and videos were captured using a Sony IMX234 at a sample rate of 30fps. XY movement tracks were extracted for each run using Noldus Ethovision 14 software, and smoothed using a one-dimensional gaussian filter (Sect. S9.1). The tdTomato lineage tracer was imaged using a standard TRITC filter cube and fluorescent light source to verify cardiac muscle cell location,

and GFPIII signal was imaged with a standard FITC filter cube to verify epidermal cell location (Sect. S9.2).

7.5 Supplementary Methods

S1. The source code:

github.com/skriegman/reconfigurable_organisms

S2. The design space.

This subsection gives bounds on the number of designs that can be built in a voxel-based workspace.

A distinct configuration of exactly N congruent cubes (voxels) connected face-to-face is known as a polycube. With N=2, there is just a single configuration: a 2-by-1 column (or dicube). But there are two perpendicular rotations: if one cube is resting on the ground, we can add the second on top or on the side.

Rotations of the same configuration can be treated as equivalent, or not. Polycubes are called "real" if equivalent under rotation, and "fixed" if not. With N=3, there are two real polycubes (tricubes), and 15 fixed. With N=4, there are eight real (tetracubes), and 86 fixed. As N increases, the number of polycubes grows exponentially. With N=16 cubes, for instance, there are on the order of 10^{10} real polycubes and 10^{12} fixed polycubes [1].

But this assumes that all cubes are identical. Here, the design space consists of polycubes composed of two cube types (passive and contractile), with any N, that fit inside an 8×8×7 bounding box. Many polycubes with N>7 won't fit inside this workspace. And with N=8×8×7=448, there is only one polycube geometry that fits, but there are still $2^{448} = 7.27 \times 10^{134}$ possible combinations of passive and contractile voxels that could be used to build the polycube.

Because the designs here are functional polycubes, their uniqueness depends on the behavioral goal. For locomotion in any direction away from the origin on a surface plane, designs can yield different behavior when inverted (Fig. 7.4), but the same behavior when the design is rotated about the vertical axis (just facing a different direction). Because there are three tissue options (none, passive, and contractile) at each point in the workspace, there are $3^{448} = 5.63 \times 10^{213}$ possible configurations, though some are isomorphic translations after reducing to a single (the largest) polycube.

S3. The physics engine.

This subsection briefly describes the physics of the in silico environment. For more details, see [95].

Experiments were performed using the voxel-based physics engine Voxelyze [95]. Voxels were connected to each other on a regular grid to form a contiguous geometry: a polycube. Interactions between adjacent voxels were modeled as flexible beams (critically damped; zeta=1) according to Euler-Bernoulli beam theory. A Coulomb friction model was applied to voxels in contact with the ground surface plane. Volumetric actuation resulting from contractile cells was simulated by oscillating the rest length between adjacent actuating voxels, in all three dimensions, when computing the elastic force between them. Additionally, a collision detection system ensured that the organisms did not self-penetrate. If a pair of surface voxels are detected to collide (intersect), a temporary beam (underdamped; zeta=0.8) was constructed between the two until the collision is resolved. A time step of 0.0032 seconds was used for numerical integration. Each design was allowed to settle under gravity for 312 timesteps (one second) before an evaluation period of 3125 timesteps (10 seconds), resulting in a total simulation time of 11 seconds.

S4. The encoding.

This subsection defines the genetic search space evolution operates in. Source code: github.com/skriegman/reconfigurable_organisms/blob/master/networks.py

S4.1. *Genotype networks.* Each configuration is genetically encoding using a Compositional Pattern-Producing Network, or CPPN [221], that maps the spatial coordinates of a 3D cartesian lattice to a single value indicating whether there is material at that location, and, if so, whether it is a contractile or passive cell. The genotype encoding is thus scale-free: A given genotype network can be mapped to arbitrary resolution coordinate space (Figs. 7.2H and S18).

We chose this particular encoding because it tends to generate spatial regularities in structure, which are known to facilitate locomotion [49]. In short, CPPNs are a special class of networks that use various activation (or interaction) functions that output regular patterns, such as sine waves and parabolas (instead of using only a single sigmoid or ramp function). CPPNs were originally proposed [221] as an abstraction of gene expression and embryonic development (rather than of neurons and brains), and we employ them as such here.

Input/output. The coordinates of each voxel are specified by their cartesian coordinates (x, y, z) and radial distances from the center of the lattice workspace (here, 8-by-8-by-7). These four coordinates (in addition to a bias term set equal to 1) are taken as input to the network. The inputs connect to (interact with) various regula-

tory and structural genes (vertices) by weighted edges (with real-valued scalar weights in the range -1 to +1) that multiply the input by the corresponding weight. Regulatory and structural genes sum all incoming weighted edges as input for their interaction function, which is here taken to be any of the following: sin(), abs(), square(), sqrt(abs()); and the negations of those four. The output expressions of regulatory genes are reweighted and passed onward to interact with additional regulatory genes (hidden nodes) or else with one of two structural genes (output nodes), whose output is thresholded at zero to determine (express) material presence and type, respectively. In some experiments (Figs. S2 and S14), an additional, independent CPPN was used to determine the phase-offset (of open-loop volumetric actuation) at each voxel.

Mutation. Both the architectures and weights of these networks are evolved: mutations add, modify or remove a single randomly-selected vertex or edge. Offspring are created by performing six kinds of mutations (add vertex/edge, remove vertex/edge, modify vertex/edge), each of which is applied with probability 1/6. If none are selected (which occurs with probability $(5/6)6 = 0.33$), one of the six is randomly chosen and applied. If a mutation is neutral (i.e., the resulting phenotypic structure is unchanged), another mutation is applied; after 1500 unsuccessful attempts at a non-neutral mutation, the 1500 neutral mutations are accepted. After mutation, the network is pruned of any erroneous edges and vertices that are not connected to the main graph (details about the evolutionary algorithm can be found in Sect. 5.1).

Initialization, hyperparameters. Networks are initialized by fully connecting the input layer to the output, and normalizing the inputs (the coordinates of the workspace) to be between 0 and 1. Then, ten random vertices are added. Next, ten randomly-selected pairs of unconnected vertices are attached by a new edge, the weight of which is drawn from a uniform distribution between -1 and 1. (If the new edge creates a cyclic graph, erase it and retry, 999 times. If 1000 failed attempts occur, terminate.) After these additions, five randomly selected edges are removed. Then, 100 edges are randomly selected (with replacement) and their weights are mutated (serially) by adding a value drawn from a normal distribution with mean zero and standard deviation, $s = 0.5$, clipping the new weight to be between -1 and 1. (Neutral mutations are permitted during this initialization process.) Finally, 100 vertices are likewise selected with replacement, and their interaction functions are replaced by functions randomly chosen from the set ±sin(), ±abs(), ±square(), ±sqrt(abs()). Hyperparameters were adopted from [124].

S4.2. *Morphological complexity.* In this work, we have demonstrated the evolution of a CPPN encoding toward various geometries (Fig. S3) and topologies (Fig. S13) capable of open-loop locomotion and object manipulation in a simulated environment. However, these shapes tend to be relatively compact and simple, lacking more complex structures, such as branching limbs.

Previous work linked morphological complexity resulting from CPPN-genotypes with environmental complexity [11]. We likewise observed the optimization of environment-specialized morphologies, such as the evolution of rudimentary end-effectors that hold an external object in place during its manipulation (Fig. S12).

To demonstrate the flexibility of the encoding to achieve more complex morphologies as dictated by the task environment, we optimized CPPNs for a single evolutionary run to generate passive structures that match target shapes (Fig. S15). To do so, we used the same evolutionary algorithm and hyperparameters (detailed below in Sect. 5) with a modified objective function: Performance was defined as the Hamming distance between binary matrices of the target and CPPN-output shapes (instead of locomotion velocity).

In a separate experiment in which designs were challenged to throw an object (Sect. 10.3), we successfully evolved functional limbs by including an objective to maximize the design's surface-area to volume ratio, in addition to an objective for thrown object distance (Fig. S14). It would be of interest to explore, in future work, the genetic encodings and behavioral selections pressures that indirectly select for complex organism geometries.

S5. Evolutionary design.

This subsection outlines the evolutionary algorithm and its objective functions, analyzes the algorithm in smaller solution spaces, and compares evolution to a gradient-based approach.

S5.1. *The algorithm.* A standard evolutionary algorithm was employed: Age-Fitness-Pareto Optimization, or AFPO [204]. AFPO uses the concept of Pareto dominance and an objective of "age" (in addition to performance) intended to promote diversity among candidate designs and prevent premature convergence. "Age" is a clade-level attribute that counts the number of generations a clade has existed in the population: Each design can trace its ancestral roots back to a randomly configured, parentless individual, which was injected into the population, with age zero, at some previous generation. Thus the age roughly corresponds to the amount of search time spent in a particular area of design space.

One hundred independent evolutionary trials were conducted, each with a unique random seed (which is here set equal to the run number, 1-100), culminating in a unique run champion: the most performant design found, according to the goal function (Figs. S2 and S3).

Each trial is initialized with a population of 50 randomly-configured designs with age zero. Every generation, the population is first doubled by creating modified copies of each individual in the population (offspring have the same age as their par-

ent). The age of each individual is then incremented by one. Next, an additional randomly-configured individual (with age zero) is injected into the population (which now consists of 101 designs). Finally, selection reduces the population down to its original size (50 designs) according to the two objectives of performance (maximized) and age (minimized). That is, starting with the youngest and most performant designs, which are by definition nondominated, successive Pareto fronts are kept in their entirety until doing so would overfill the population past its original size (50 designs), at which point designs are added stochastically with probability proportional to their performance. This process of random variation and directed selection is repeated for 1000 generations.

A third objective was used in addition to performance and age: number of contractile voxels (minimized). This additional objective was added to ensure organisms had sufficient non-functional volumes that could be replaced by future non-actuating building blocks which may be required for specific tasks, such as wavelength perception (opsin-like detection), or metabolic pathways necessary for nutrient uptake and consumption. In spite of this objective, none of the run champions from the first pass for locomotion utilized passive voxels (Fig. S2).

Similarity to other architecture search algorithms. As mutations not only tune the parameters of an existing (parent) network, but can also add and remove genotype network structure (edges and vertices), the evolutionary algorithm is performing what is known as "architecture search" [69]. Many evolutionary approaches to architecture search exist, such as NEAT (NeuroEvolution of Augmenting Topologies; [222]). However, NEAT evolves artificial neural networks, whereas we are evolving bodyplans. More specifically, we are evolving CPPNs (genotype; Sect. 4.1) that encode bodyplans (phenotype). HyperNEAT (Hybercube-based NeuroEvolution of Augmenting Topologies; [223]) is an extension of the NEAT method which evolves CPPNs that typically, as the name suggests, encode neural network connectivity patterns. However, HyperNEAT has also been used to evolve CPPNs that encode soft robot bodyplans [42]. We chose to evolve CPPNs using AFPO instead of HyperNEAT because the former is a much simpler algorithm than the latter. Despite the fact that it is more complex, it would be interesting to apply HyperNEAT to designing reconfigurable organisms; it could be that the additional machinery yields more performant designs.

Source code: github.com/skriegman/reconfigurable_organisms/blob/master/tools/algorithms.py

S5.2 *Measuring performance.* In all experiments, designs are allowed to settle under gravity for one second before the evaluation period begins. Just before the evaluation period starts, the initial center of mass of the design is recorded as (x_0, y_0, z_0).

For locomotion, the performance score was net displacement of the design's center of mass, in terms of euclidean distance: the square root of $(x - x_0)^2 + (y - y_0)^2$, where (x, y) is the final position of the design on the ground plane at the end of an evaluation

period of 10 seconds. For object manipulation (Sect. 10.1), transport (Sect. 10.2), and expulsion (Sect. 10.3), the object's net displacement was tracked instead of the design's.

It is important to note that the formulation of the performance objective function must in some cases be refined in order to realize desired behavior. For example, an early version of the objective function for object transport (Fig. S13) intended to reward how far an object could be carried. However, because the lightweight object never touched (i.e., penetrated) the simulated ground plane, the optimizer discovered designs that dragged the object along the ground plane. This was corrected by constraining the object to be completely surrounded by tissue (Sect. 10.2).

S5.3 *Runtime.* Each evolutionary trial optimized a population of 51 designs on a dual-processor, 12-core Intel E5-2650 v4 (i.e., 24 threads). No trial took less than 18 wall-clock hours (432 CPU hours) or more than 22 wall-clock hours (528 CPU hours) to evaluate 1000 generations of evolutionary improvement. The runtime varies because the algorithm is stochastic: some designs have more voxels than others and thus require more CPU-time. Because evaluating designs in simulation is the computational bottleneck, the algorithm is readily parallelizable: doubling CPU threads halved the wall-clock time (10 hours when tested using two Intel E5-2650 v4s).

S5.4 *Algorithm analysis.* It is difficult to know what the optimal design is for large search spaces. So, we investigated a search space in which it was possible for us, given our computational resources, to determine exactly what the optimal design is (for a given random actuation pattern). We started with a 2×2×2 workspace and identified the optimal design for five different, random actuation patterns (Fig. S17A). The evolutionary algorithm found the optimal design for all five actuation patterns in 9129 evaluations (179 generations). In the slightly larger, 3×2×2 workspace, the evolutionary algorithm found the optimal design for all five actuation patterns in 4284 evaluations (84 generations) (Fig. S17B). In a 3×3×2 workspace, the algorithm took much longer, requiring 113,628 evaluations (2228 generations) to find the optimal design for all five actuation patterns (Fig. S17C). We could not determine the optimal design at 3×3×3, so we terminated the determination of optimal designs at this point. Source code: github.com/skriegman/reconfigurable_organisms/blob/master/exp/Algorithm_Analysis_EA.py

S5.5 *Comparison to a gradient-based approach.* We employed an evolutionary algorithm because we have considerable experience using this particular algorithm to evolve soft robots in previous work. However, there are many forms of constrained optimization. One of the most common is stochastic gradient descent. So, we applied a stochastic gradient-based method (Parameter-exploring policy gradients; [205]) directly to the design problem. Typically the policy that is optimized is a vector of neu-

ral network weights (floating-point values). Here, the policy is a static arrangement of the discrete building blocks (ternary values). The algorithm samples the space of designs, evaluates their performance in silico, and estimates the gradient using a popular stochastic gradient descent optimizer (Adam; [112]). This algorithm yielded a less performant design (4.6 body lengths per minute; Fig. S16) that failed all three conditionals of the build filter (Sect. 7.2). Source code, which was adapted from [88], is available here: github.com/skriegman/reconfigurable_organisms/blob/master/exp/Algorithm_Analysis_SGD.py

The same population size as the evolutionary algorithm was used; all other hyperparameters were left at their default values as reported in (46). The policy vector of floating-point values was discretized to determine tissue type at each point in the workspace: values below -0.05 were left empty, values above 0.05 were encoded as passive voxels, and values in between -0.05 and 0.05 were encoded as contractile voxels. The two thresholds of -0.05 and 0.05 were set such that randomly-initialized designs at the beginning of optimization contained all three tissue types (the initial standard deviation is 0.10 by default). This gradient-based algorithm quickly converges to suboptimal designs, even in very small search spaces (Fig. S17). To combat this premature convergence, we modified the algorithm to restart, every 1000 evaluations, from a different random initialization.

After 10,000 evaluations of search in the 2×2×2 workspace, the policy gradient algorithm found the optimal design for two of the five, random actuation patterns (40%). By restarting every 1000 evaluations, the modified algorithm was able to find the optimal design for four out of the five actuation patterns (80%) (Fig. S17A). In the 3×2×2 workspace, the algorithm did not find the optimal design for any of the five actuation patterns (0%). Modified to restart every 1000 evaluations, the algorithm found the optimal design for one of the five actuation patterns (20%) (Fig. S17B). In the 3×3×2 workspace, we allowed optimization to continue for much longer (140,000 evaluations), but the optimal design was not found for any of the five actuation patterns, with or without restarts (Fig. S17C). Source code: github.com/skriegman/reconfigurable_organisms/blob/master/exp/Algorithm_Analysis_SGDre.py

S6. Updating design constraints.

This subsection describes how the design process is improved with feedback from the behavior of manufactured organisms.

On the first pass through the pipeline using the goal behavior of locomotion, the simulated environment consisted of an infinite plane and a gravitational acceleration of -9.81 m/s^2. Both the passive and contractile building blocks had a Young's modulus of 107 Pa, a Poisson's Ratio of 0.35, and coefficients of static and dynamic friction

of 1.0 and 0.5, respectively. Source code:
github.com/skriegman/reconfigurable_organisms/blob/master/exp/Locomotion_pass1.py

The contractile voxels were volumetrically actuated (±50% rest volume) at 5 Hz. Although the cardiac tissue used to build reconfigurable organisms can only contract, simulated actuation also expanded voxels in volume because this produces more force, and thus faster locomoting designs in silico. We chose to call these voxels "contractile" to help clarify the match with contractile tissue. Future design-manufacture cycles of the pipeline could add a contraction-only actuation constraint in silico, however it was not necessary for successful transferral of behavior (Sect. 9.1).

The phase-offsets of actuation for each voxel (from a global, sinusoidal signal) were co-optimized with morphology, using two independent genotype networks (Sect. 4.1). Designs evolved propagating waves of volumetric actuation, yielding rapid locomotion via bounding gaits, with timeframes (on average, 47% of the gait cycle) in which no part of the design was in contact with the simulated ground plane (Fig. S2). However, when these designs were manufactured in vivo, they always kept part of their ventral surfaces in contact with the surface of the dish due to negative buoyancy.

This discrepancy was rectified by adding the following constraints to the simulated environment and actuation. On the second pass through the design pipeline, an aqueous environment was simulated by decreasing the gravitational acceleration to -0.1 m/s^2, and applying a drag force to surface voxels, assuming a fluid density of water ($\rho = 1000$ kg/m^3) and a drag coefficient of $C = 1.5$ for each exposed voxel face.

Voxels were simulated with half the Young's modulus (5×10^6 Pa) and five times the length scale of those used during the first pass of the pipeline. This allowed a larger numerical integration timestep to be stable (see [95] for details), which greatly reduced the required CPU time of each evaluation.

Actuation frequency was reduced to 2 Hz to remove momentum effects (which are difficult to simulate accurately) and to better match the contraction rate of the cardiac tissue (∼1 Hz). The phase-offset of each voxel was randomized instead of optimized, which prevents designs from overfitting to a specific actuation policy. Each randomly-configured design injected into the population (with age zero; see Sect. 5.1) was assigned 448 phase-offsets, randomly drawn from a uniform distribution between $-\pi/2$ and $\pi/2$, one value for every point (possible voxel location) in the 8×8×7 workspace. These phase-offsets were then hardcoded for the entire clade.

Evolutionary improvement within a clade thus occurred through changes in overall shape, and distribution of the passive and contractile voxels, to collectively derandomize the global movement produced by the random actuation. This reduced the dependence on precisely-timed actuation, which increased the likelihood of successful transferal from silico to vivo. Source code:
github.com/skriegman/reconfigurable_organisms/blob/master/exp/Locomotion_pass2.py

The surface contact behavior of designs generated on the second pass (Fig. S3) better matched that of the actual living systems: On the second pass through the design pipeline, run champions were, on average, in contact with the ground plane for 93.3% of their evaluation period, compared to just 52.7% on the first pass. The proportion of simulation time that designs are in contact with the ground was computed by recording (100 times per second) whether or not the z position of any voxels were penetrating the surface plane, using the following source code:

skriegman/reconfigurable_organisms/blob/master/data_analysis/Time_in_contact_with_ground.py

S7. Filtering evolved designs for manufacture.

This subsection describes how designs were filtered based on their performance in silico, transfer potential, manufacturability under the current build method, and their scalability to more complex tasks in future deployments.

S7.1. *The robustness filter.* The most performant designs (Fig. S3) were sorted by their robustness to random perturbations in their actuation (Fig. S4). Each design was then copied 20 times. The phase-offsets of the actuating voxels in each copy were independently mutated by adding a number that was drawn randomly (with random seeds 1-20) from a normal distribution with mean zero and standard deviation $s = 0.4\pi$ (which is 40% of the $-\pi/2$ to $\pi/2$ range of valid phase-offset values). This hyperparameter was selected to be large enough to scramble the original phase-offset value without being so large as to push all mutations up against the $\pm\pi/2$ bounds. Designs that maintained the highest average performance across this actuation noise were passed, one by one, in order of their robustness ranking, to the build filter. Source code:

github.com/skriegman/reconfigurable_organisms/blob/master/data_analysis/Robustness_Filter.py

S7.2. *The build filter.* The most robust designs were evaluated by their manufacturability under the current build method, which layers contiguous tissue regions sequentially, one on top of the other, with each layer filling the x,y plane (Fig. S6; Sect. 8.2). Thus, the first criterion a design must meet for fabrication is that it must contain contiguous tissue regions that fill the dorsal and ventralmost x,y planes with just one of the two tissue types (i.e., no mixing within the dorsal and ventral layers). The design can be rotated in 3D space to satisfy this criterion (e.g., without mixing within the anterior and posteriormost layers).

Secondly, designs cannot contain arbitrarily small gaps in their geometry, because they are made of differentiating cells which will adhere to neighbors if they come into contact with each other. The minimal concavity was examined by producing organisms with progressively smaller shape deformations, then determining which persist

across the lifespan of the organism, and which close due to tissue adherence/contraction, leading to loss of concavity. Preliminary work determined that concavities with a width of 100μm or greater produced stable long term deformations suitable for biological building (Fig. S7). As the organisms typically have body diameters in the range 750$\pm$100μm at the time of initial cutting and 850$\pm$100μm after four days of healing, the minimal concavity width is 12%-14% of total body length.

Because larger designs can travel farther in the same amount of time as smaller ones, the in silico designs tend to fill at least one horizontal dimension of the 8×8×7 workspace (Fig. S3), resulting in a maximum length of 8 voxels. This equates to a minimal concavity width of a single voxel (1/8=12.5%). Thus, the second criterion for fabrication is that the design cannot contain gaps less than two voxels wide.

Finally, more complex tasks can require room for payloads (Fig. S13; Sect. 10.2) or, in future, the addition of specialized cells for purposes other than locomotion, including sensory input, metabolism, memory, biosensors, etc. Also, contractile tissue incurs a high metabolic cost (compared to non-muscle tissue), which decreases the total lifetime of transferred designs. Thus, the third criterion for fabrication is that the design must be mostly (more than 50%) passive. Source code:
github.com/skriegman/reconfigurable_organisms/blob/master/data_analysis/Build_Filter.py

S8. Manufacturing reconfigurable organisms.

This subsection summarizes the current build method.

S8.1. *Cilia-driven organisms.* During preliminary experiments aimed at testing the feasibility of in vivo designs, the manufactured organisms lacked contractile tissue and were instead propelled by cilia present on the surface of the body. These cilia-propelled spheres were manufactured in exactly the same way as the cardiomyocyte-driven organisms used to measure transfer success (Sect. 8.2), except the Notch intracellular domain (Notch-ICD) was not overexpressed through synthetic mRNA microinjection (this step inhibits multiciliated cell formation) prior to building. However, all of the ciliated organisms moved when released into the aqueous environment (albeit in unpredictable directions and at unpredictable speeds). Thus, measuring transferability between in vivo and in silico designs could produce false positives. In addition, the accurate modeling of swimming and fluid dynamics proved to be challenging in simulation, so we altered our build method to produce contractile based movement.

S8.2. *Cardiomyocyte-driven organisms.* Contractile organisms were generated using two separate approaches (detailed in Methods and Materials). In the first, presumptive cardiomyocyte and epidermal cells were extracted from embryos and dissoci-

ated in calcium free magnesium free medium. Dissociated cells were then transferred to a 1mm depression and the tissues were layered according to the desired in silico design. After two days of further development, final shaping was performed using a microcautery device and surgical forceps. For the second build method, the presumptive cardiomyocyte and epidermal tissues were not dissociated to the individual cell level. Instead, layers of tissue were stacked on top of one another, with the cardiomyocyte layers in the center. Shaping occurred in exactly the same manner, using a combination of microcautery and surgical forceps. Both the dissociation and tissue layering method were employed during the course of the study, however, we chose to focus on the latter for movement-based assays. While this method results in lower accuracy of tissue placement (as specific numbers of cells could not be layered with precision), it significantly reduces total build time, allowing the investigator to produce approximately ten times the amount of organisms per unit time compared to the dissociation method.

S9. Measuring transfer success.

This subsection explains how the behavior and structure of manufactured organisms were compared to those of the in silico design.

S9.1. *Transferal of behavior.* The design was evaluated 25 times, resulting in 25 movement trajectories. Each time, the evolved set of phase-offsets for the actuating voxels was different, resulting in slightly different behavior (Fig. 4). This was done by adding to each of the original phase-offsets, a value that was drawn randomly from a normal distribution with mean zero and standard deviation $s = 0.4\pi$ (which is 40% of the $-\pi/2$ to $\pi/2$ range of valid phase-offset values). This hyperparameter was selected to be large enough to scramble the original phase-offset value without being so large as to push all mutations up against the $\pm\pi/2$ bounds.

The numerical integration step size was 0.0032 seconds. For each random actuation pattern, the design was allowed to settle under gravity for 312 timesteps (one second), and then evaluated for 18750 additional timesteps (60 seconds), resulting in a total simulation time of 61 seconds.

Six reconfigurable organisms were built which embodied this design. They were imaged live for 10 minutes in 0.75x MMR at 20°C using a Nikon SMZ-1500 microscope equipped with both top and substage illumination. Videos were captured using a Sony IMX234 at a sample rate of 30fps. Behavioral trajectories were extracted using Noldus Ethovision 14 software. Each trajectory was then smoothed using a one-dimensional gaussian filter with a kernel standard deviation of 30 seconds.

Statistical analysis. In measuring transfer success, we made three comparisons

(controlling for fals discovery rate; (47)):

1. movement heading in vivo relative to in silico;
2. net displacement in vivo upright relative to inverted; and
3. net displacement in silico upright relative to inverted.

For movement heading, the data consist of the dichotomous outcomes: either the designed organism moved in the predicted direction, or not. Although organisms could move in any direction (0 to 360° relative to the predicted heading) we discretize the space into four directions of possible movement (forward, backward, left, right), only one of which is considered a success (forward). We have six realizations of the design in vivo (six separate organisms) that were each reset to the origin four times. As resets to the origin are clearly not independent observations, $n = 6$. That is, we have six independent Bernoulli trials with probability of success q. Because there are four directions of possible movement, the null hypothesis is that organisms move in the predicted direction with probability $q = 1/4$. All instances moved in the predicted direction, though one organism (and its four resets) was more or less sessile (with small amounts of movement in the predicted direction). The most conservative use of the data is to consider only five of the six realizations to be successful, and thus $p = 4.6 \times 10^{-3}$. Controlling for false discovery rate (47), the null hypothesis is rejected at the 0.01 level of significance.

For net displacement, the data consist of paired replicates: anatomically-upright ("pretreatment") and anatomically-inverted ("posttreatment") observations from the same individual design. We are concerned with a shift in location (i.e., the median of the distribution of net displacement) due to inverting the design (the application of the "treatment"). We use a distribution-free signed-rank test (Wilcoxon). The null hypothesis asserts that each of the distributions for the differences (posttreatment minus pretreatment observations) is symmetrically distributed about 0, corresponding to no shift in location due to the treatment (inverting the design). In silico, $p = 9.7 \times 10^{-5}$. Controlling for false discovery rate, the null hypothesis is rejected at the 0.001 level of significance.

In vivo, four organisms were evaluated five times while upright, and five times inverted. However, the recording equipment failed during one of the upright runs (Trial 2) for three of the organisms. This trial was simply removed from consideration for these three (i.e., displacement while inverted in Trial 2 was discarded where corrupted). The two other organisms were evaluated five times upright, but only once while inverted since these organisms generated little to no displacement when inverted, and motion tracking resources were limited. The single evaluation while inverted was therefore used as the posttreatment observation for each of the six upright (pretreatment) observations. In vivo, $p = 2.6 \times 10^{-5}$. Controlling for false discovery rate, the null hypothesis is rejected at the 0.0001 level of significance.

In fact, only one of the six organisms was observed to produce appreciable forward movement while inverted. This anomaly was likely due to the cardiac tissue being layered deeper (more dorsally) than the other designs, resulting in a small amount of deformation on the dorsal surface.

Source code for reproducing the in silico behavioral trajectories: github.com/skriegman/reconfigurable_organisms/blob/master/data_analysis/Transferal_from_silico_to_vivo.py

Source code for the statistical analysis: data_analysis/Statistical_Analysis.py

S9.1. *Transferal of structure.* The structure of organisms was compared to that of the in silico design using 2D images and Hausdorff distance. To quantify the structural error (Fig. S8), lineage labeled organisms were created by harvesting tdTomato expressing cardiac progenitor tissue from one set of donor embryos, and GFPIII expressing passive epidermis tissue from a second set of donor embryos. Donor tissue was dissociated and transferred to 1mm concave wells, placing both cell types in proportions and locations matching the in silico design. Two days later, the organism was then sculpted to the desired shape, and imaged in multiple orientations using a FITC filter set to visualize epidermal tissue (green) and a TRITC filter set to visualize cardiac progenitor (red) tissue. Channels were then overlaid in ImageJ to create the final composite image for analysis. K-means clustering was used to classify each pixel as one of three tissue types: contractile, passive, or none.

The Hausdorff distance between in vivo and in silico pixels was calculated for passive tissue only, then again for contractile tissue. The closest matching pixel (i.e., with the same tissue type) was found in silico for all in vivo pixels, and the closest matching pixel was likewise found in vivo for all in silico pixels. The Hausdorff distance is the largest such discrepancy between vivo and silico tissue coordinates, in terms of euclidean distance in microns. A small Hausdorff distance indicates that for every pixel in vivo there is a pixel of the same tissue type nearby in silico, and vice versa. After measuring the Hausdorff distance for both tissue types, the larger of the two is taken to be the structural error.

Formally, the Hausdorff distance for tissue type 1 is defined as:

$$H_1 = \max\{ \sup_{s \in S_1} \inf_{v \in V_1} d(s,v),\ \sup_{v \in V_1} \inf_{s \in S_1} d(s,v) \},$$

where S_1 and V_1 are the sets of in silico and in vivo pixels that were classified as the first tissue type, and $d(s,v)$ is the euclidean distance between pixels s and v. Similarly, the Hausdorff distance for tissue type 2 is defined as:

$$H_2 = \max\{ \sup_{s \in S_2} \inf_{v \in V_2} d(s,v),\ \sup_{v \in V_2} \inf_{s \in S_2} d(s,v) \},$$

where S_2 and V_2 are the sets of in silico and in vivo pixels that were classified as the second tissue type. The structural error is then: $\max(H1, H2)$.

Before these comparisons can be made, however, the in vivo and in silico images need to be on a relatively consistent coordinate frame. Thus, images were auto-cropped such that the edge of the design fills the frame. This is done by converting each image to grayscale, and thresholding at 10/255 to create a binary image, thus isolating the organism against the dark background. Contours are then automatically drawn on the image by traversing the boundaries of each transition between black and white to find closed loops (each of which is a contour). A bounding box is drawn around the largest contour (by area), and the rest of the image is trimmed off. Finally, the cropped images were resized (downsampled) to a constant resolution (50×50).

A grid search was then performed to find the 3D perspective with the lowest structural error. The design was lowered/raised in elevation angle in the z plane, and spun around by azimuth angle in the x,y plane, in increments of 10° in each dimension. Then the 3D plot was saved as a 2D image and rotated again in increments of 10°.

Since the tissue regions are classified via unsupervised clustering (and were thus assigned arbitrary labels), we take the regions with the largest overlap to be of the same type. This introduces the possibility of similarly shaped but different tissue types aligning to achieve the lowest structural error for a given in vivo perspective, but this did not occur in our experiments. However, the rotation with the lowest structural error did not always respect the organism's anteroposterior alignment (the design was sometimes "facing" the opposite direction). So we restricted rotations of the design to better match the range of perspectives captured across the four images of the organism: the grid search was constrained to elevation angles between -40 and 60°, azimuth angles between -120 and -60°, and rotations of the resulting 2D image between 0 and 30°.

The average structural error achieved across all four images was 323 microns (38% of the organism's largest diameter) (Fig. S8).

Note that this preliminary method does not account for distortions resulting from flattening a 3D object to a 2D image (e.g., the moon terminator illusion). Future work will aim to capture many images of the organism from different perspectives, and use them to virtually reconstruct a 3D model for direct structural comparison. Source code: github.com/skriegman/reconfigurable_organisms/blob/master/data_analysis/Structural_Error.py

S10. Applications.

This subsection summarizes four additional goal behaviors in which organisms can interact with an external object or other organisms.

S10.1. *Object manipulation.* Introducing a single object to the environment, and altering the goal function to track the external object, instead of the design, yielded object manipulation—block pushing—in silico (Fig. S12). The new behavioral goal

input into the pipeline was to maximize displacement of a 2×2×2 voxel object during an evaluation period of 30 seconds. This extended evaluation time of 30 seconds (instead of 10 sec) prevented the strategy of simply falling onto the object and hitting it forward ahead of the (often immobile) design. All other constraints from the second pass for locomotion were left in place: passive and randomly-actuating building blocks were reconfigured within an 8×8×7 workspace, and evaluated in a floored aqueous environment. Sixteen independent evolutionary runs were performed. In some designs, a primitive end-effector—a notch in the corner of the body—evolved to hold and manipulate the object as it was pushed along the floor. Source code: github.com/skriegman/reconfigurable_organisms/blob/master/exp/Object_Manipulation.py

S10.2. *Object transport.* Some designs evolved for locomotion were hollow and could thus, in theory, be exapted to internally store the 2×2×2 voxel object as cargo, rather than push it externally. This realization led to the new goal of maximizing the euclidean distance of a carried object. Using a slightly larger, 10×10×9 workspace, a mask was placed on the design requiring the morphology to house the 2×2×2 object within a 4×4×4 voxel pouch. This yielded evolved object transport in silico (Fig. S13). All other constraints were identical to the second pass for locomotion (the evaluation period was reverted back to 10 sec). Sixteen evolutionary runs were performed.

Some modifications of the objective function were required to realize object transport. An early version of the objective function did not use masking to force the object to be inside the design at the center of the workspace. The object was instead free to be positioned anywhere in the workspace. Designs were then rewarded by net object displacement, with the stipulation that the evaluation period would be terminated if and when the object was detected to penetrate the floor. This is a standard way to ensure that a simulated object is touching the ground. However, given the buoyancy of the aqueous environment, the lightweight object would often touch but not penetrate the ground. What evolved were mostly variants that pushed or dragged the object across the ground plane. Source code: github.com/skriegman/reconfigurable_organisms/blob/master/exp/Object_Transport.py

S10.3. *Object expulsion.* A non-locomotion-based goal was also supplied to the pipeline: maximize thrown object distance. In other words, evolve a catapult [40]. We dropped a 2×2×2 voxel object (the projectile) from one voxel length above the 8×8×7 workspace, but no designs evolved to catch and throw the object forward. So we reverted actuation to be finely tuned (rather than randomized) by co-evolving phase-offset alongside structure, as in the first pass for locomotion. But, the evolved designs were relatively compact and lacked limbs for proper throwing. So, we induced direct selection pressure for limbs by incorporating an additional objective that maximized

surface-area to volume ratio. In three of five evolutionary trials, limbs evolved which the design employed to throw the object (Fig. S14). However, all of the designs capable of throwing objects, failed to pass through the build filter (conditional B1 of the flowchart in Fig. S4) because they were composed entirely of muscle. Additionally, these designs rely on finely-tuned actuation, which is unlikely to transfer from silico to vivo. Source code:

github.com/skriegman/reconfigurable_organisms/blob/master/exp/Object_Expulsion.py

S10.4. *Collective behavior.* Multiple designs can be placed in a single environment instead of a single one. This can be done by simply relaxing the constraint which takes only the largest connected component to be the design. Alternatively, individually evolved designs can be evaluated together at a later stage in the pipeline. The latter method was implemented: five of the fastest locomoting designs (Fig. S3) were placed in the same environment amongst a grid of particulate matter in the form of 2×2×1 voxel particles, yielding spontaneous collective behavior and particle aggregation in silico (Fig. S10). Source code:

github.com/skriegman/reconfigurable_organisms/blob/master/exp/Collective_Behavior.py

Spontaneous collective behavior and aggregation of external particles also occurred in vivo when multiple (10 to 15) organisms were placed together at the center of a single petri dish containing carmine dye (Sigma-Aldrich C1022-5G) (Fig. S11). A stock solution of carmine dye was created at a concentration of 0.01g per 10ml 0.75x MMR and vortexed for 10 seconds. Individual working solutions were then created in 1% agarose coated polystyrene petri dishes by diluting the stock 1:10, again in 0.75x MMR, for a final concentration of 0.001g per 10ml. Dishes containing the working solution were housed under an imaging microscope and allowed to settle for four hours at 22°C, creating a layer of particulate dye on the surface of the dish.

As a control, organisms were withheld from the dish, and the dye did not self aggregate after 1 or 24 hours.

S11. Scaling the pipeline.

This subsection describes how future design-manufacture pipelines may improve to scale the complexity and competence of reconfigurable organisms.

In the work reported here, actuation in silico was constrained to be random (Sect. 6) because it is not yet understood how to model the dynamics of cells in novel configurations and environments. This proved sufficient for some open-loop tasks, but not for others which required finer control (Sect. 10.3). Before a general understanding of the relationship between cell signaling and behavior is realized, behavioral biases of cells, or correlations between cells in certain configurations could be

captured and distilled down into new constraints. However, in order to accurately incorporate the constraints of more realistic dynamics, the resolution of the simulation might need to be improved in future experiment designs.

Designs were constrained to 8-by-8-by-7 resolution (448 voxels) but the cardiomyocyte-driven organisms contained 5,000 cells (Fig. S18). The resolution of the workspace could be increased, but the required CPU-time scales with $O(n + s^2)$, where n is the total number of voxels, and s is the number of voxels that lie on the surface of the body, which are used to detect and resolve collisions. Such computational bottlenecks can be circumvented by incorporating an additional generator-filter pair along the pipeline that draws individuals from an evolving population of low-resolution designs (and/or low-resolution task environments) to evaluate at higher-resolution. In effect, the low-resolution population would serve as a computationally-cheaper surrogate model for higher-resolution designs and environments. However, it is currently not known how to guarantee that behavior generated in low resolution design space can be preserved when mapped onto higher resolution spaces.

The CPU-based physics simulator used in this work rendered repeated evaluation of very high resolution designs, or the collective behavior of a swarm in silico computationally intractable. This was because the forces and positions of each cell (voxel) within each simulated design were calculated sequentially rather than simultaneously. CPUs have tens of threads on which they can carry out independent processes in parallel, though most physics simulators use just a single thread. GPUs, in contrast, contain hundreds of thousands of threads.

A GPU-accelerated version of the physics engine used throughout this thesis was recently released [141]. This raises the possibility of simulating reconfigurable organisms at the cell level, and exporting the design directly to a 3D bioprinting/cell-printing device containing multiple dispensers (each loaded with a bio-ink of an individual cell type), which could in principle exactly recapitulate the design in vivo. However, much work remains to adapt existing technologies for this purpose.

Video S1: Designing reconfigurable organisms.

Link: youtu.be/VVd_MjHm_tc

Video S1 first previews successful output of the pipeline: the champion design of run 52 in silico and its realization in vivo. Next, the input—the goal (move into the left-hand side of the screen, as fast as possible) and building blocks (passive and contractile voxels)—are introduced. The building blocks are shown in pairs (two passive, cyan blocks are connected and positioned on the floor next to a separate pair of contractile, red/green blocks) to better illustrate the actuation dynamics. At 0:30, three random configurations of the building blocks are shown in action. These three

configurations were the best designs (of 50 random designs) found during generation zero of three independent evolutionary runs with goal behavior of net displacement. One thousand generations later, the most robust design to pass through the build filter—the champion of run 52—is selected for construction in vivo. At 1:14, video of the virtual design (top) is complemented by video of its realization in vivo: a computationally-designed organism locomoting with contractile muscle in an aqueous environment (presented at 4x speed). At 1:27, an organism is shown to move in the direction predicted by the in-silico model (one second of video corresponds to 1.25 real-time minutes). At 1:37, the organism is rotated 180° about its transverse plane (placed on its back) and fails to generate forward movement. At 2:12, the collective behavior of multiple designs were deployed in an aqueous environment and filmed in timelapse. Individuals are observed forming stable duplets and triplets, changing partners across the observation. This behavior was predicted and observed by the in-silico model (particularly at 2:20). At 2:28, the organism is deployed into an aqueous environment with an agarose substrate covered in carmine particles, and spontaneously pushes one of the particles forward through a circling movement pattern. A design explicitly evolved in silico for pushing a particle (yellow) is also shown for comparison. At 2:39, a design evolved for locomotion is exapted for object transport (yellow). Following a demonstration of the brittleness of our current technologies, at 3:02, an organism was subjected to mechanical laceration with microsurgery forceps. Timelapse imaging demonstrates wound closure and repair over the course of 10 minutes.

Video S2: Manufacturing reconfigurable organisms.

Link: youtu.be/kCOKtmmNH90

Video S2 provides a timelapse recording of the entire build process for in vivo representations of the in silico designs. The process begins with the microinjection of fertilized Xenopus embryos (1.15-1.2mm in diameter) with synthetic mRNAs encoding lineage tracers (tdTomato [red] or GFPIII [green]) or proteins to alter cell fate. At 0:10, after developing for 24 at 14°C, the vitelline membrane is removed from each embryo with microsurgery forceps. At 0:23, the animal cap of each embryo is then surgically excised with forceps and incubated for 10 minutes in dissociation media. At 0:36, the outer ectodermal layer (which does not dissociate) is manually separated from the inner cell layers (approximately 30μm in diameter). The remaining tissue is gently agitated to further aid with dissociation, before being pooled with a micropipette (1:08). At 1:20, hundreds of cells and at 1:27, thousands of cells are transferred to agarose wells containing standard media (different cell types can be layer sequentially at this step), promoting cell re-aggregation. At 1:34, cell aggregates undergo compaction as they re-adhere and are allowed to develop for two additional

days at 14°C. At 1:50, a microcautery device is utilized to define the rough shape of the transferred design (650μm-750μm in diameter) and microsurgery forceps are then used to sculpt the final features.

Table S1. Biological reagents.

1	*Xenopus* frogs and embryos:	Nasco LM00490, LM00531, LM00456
2	Standard Xenopus media (10x stock):	1M NaCl, 20mM KCl, 10mM $MgSO_4$, 20mM $CaCl_2$ dihydrate, 20mM HEPES, ph 7.4.
3	0.75x Xenobot media:	Diluted from 10x Stock Xenopus media
4	Cell dissociation media:	50.3mM NaCl, 0.7mM KCl, 9.2mM Na_2HPO_4, 0.9mM KH_2PO_4, 2.4mM$NaHCO_3$, 1.0mM EDTA, pH 7.3.
5	tdTomato lineage tracer:	Waldner C, Roose M, Ryffel GU. 2009. BMC Dev Bio. 9(1):37.
6	GFPIII lineage tracer:	Zernicka-Goetz M, et al. 1996. Development. 1;122(12):3719-24.
7	NotchICD cilia inhibiting construct:	Deblandre GA, et al. 1999. Development. 1;126(21):4715-28.
8	1% agarose substrate for cap movement trials:	VWR EM-2125
9	Petri Dish, 60mm x 15mm polystyrene:	Fisher FB0875713A
10	Carmine dye:	Sigma-Aldrich C1022-5G
11	Gentamicin:	ThermoFisher Scientific, 15710072

Table S2. Shaping tools.

1	Microsurgery forceps:	Dumont, 11241-30 #4
2	Forcep sharpening stone:	Fine Science Tools 29008-22
3	Microcautery device:	Protech International Inc., MC-2010
4	13 micron cautery electrode:	Protech International, 13-Y1
5	Binocular dissecting microscope:	VWR 19000-852

Table S3. Imaging.

1	Fluorescent microscope:	Nikon SMZ-1500
2	Microscope camera:	QImaging Retiga 2000R CCD
3	Ipod touch for time lapse imaging:	Apple Inc.
4	Ipod microscope mount:	Gosky Binocular adapter B013D2ULO6
5	ImageJ image analysis software:	National Institutes of Health
6	Motion tracking software:	Noldus Ethovision 14

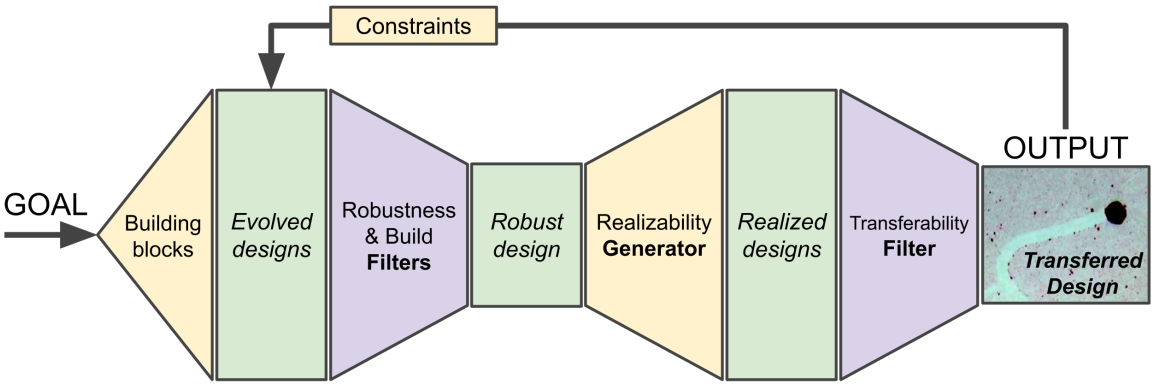


Figure S1. **Pipeline diagram.** The pipeline goes from left to right—from input goal to in-silico design to in-vivo output, and back. The pipeline continues to loop indefinitely, or until some external termination condition is reached.

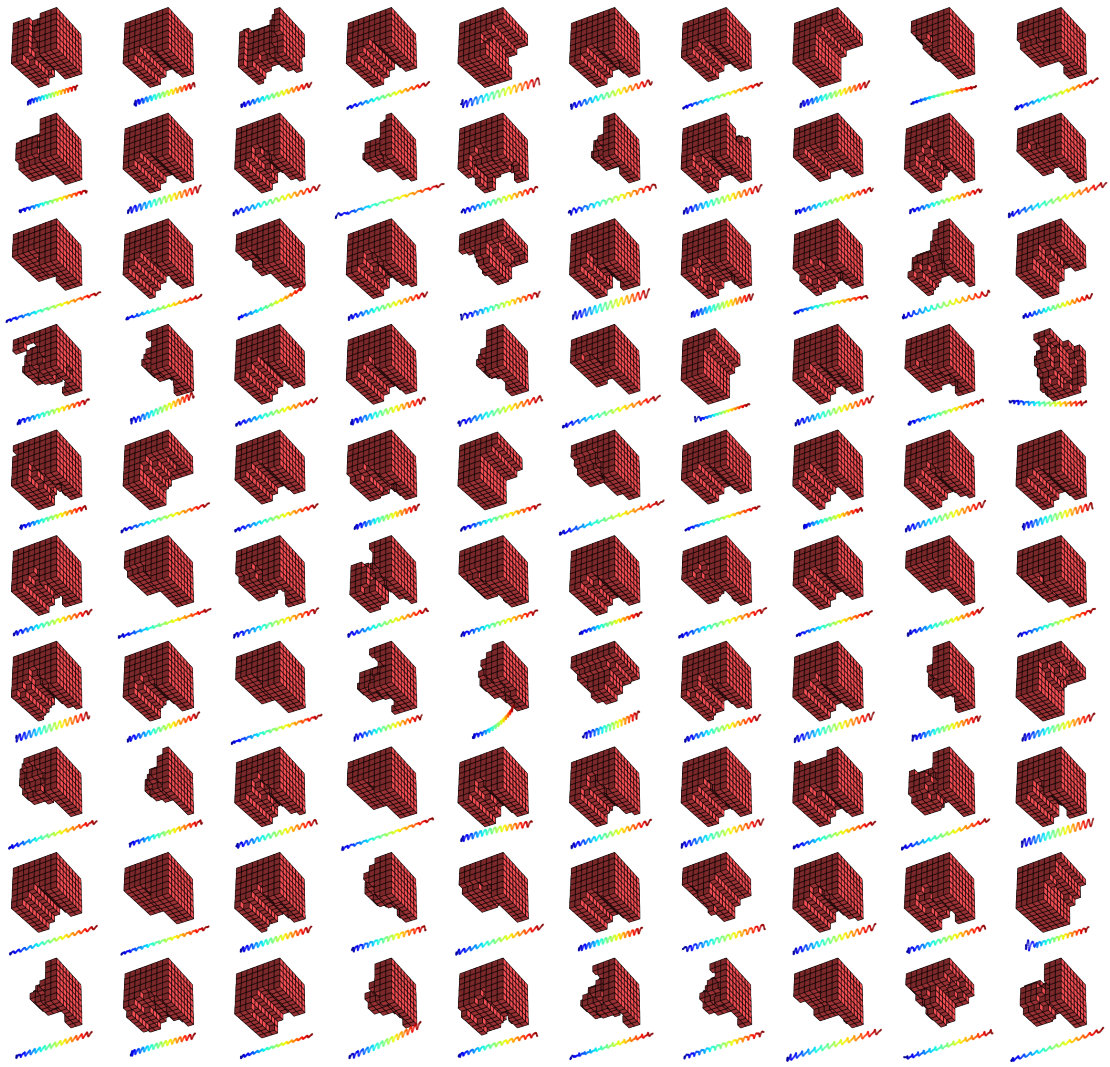


Figure S2. **The fastest designs from the first pass of the pipeline for locomotion.** During the first pass of the pipeline for the goal behavior of locomotion, designs were evaluated in an Earth-like terrestrial environment and evolution was permitted to finely tune the phase-modulation (from a global signal) of actuation in each voxel. Under these conditions of precise actuation, and despite an objective explicitly minimizing the number of contractile voxels (red), the best designs (defined by their locomotive ability) did not utilize any passive material. There was also a strong convergence in shape: many independent runs converged to similar geometries. The rainbow streak below each design is its behavior—from initial (blue) to final position (red)—, superimposed, shifted and scaled to fit on the image (each streak is identically scaled). The run champions for runs 1 through 100 (with random seeds 1-100) are pictured in row-major order.

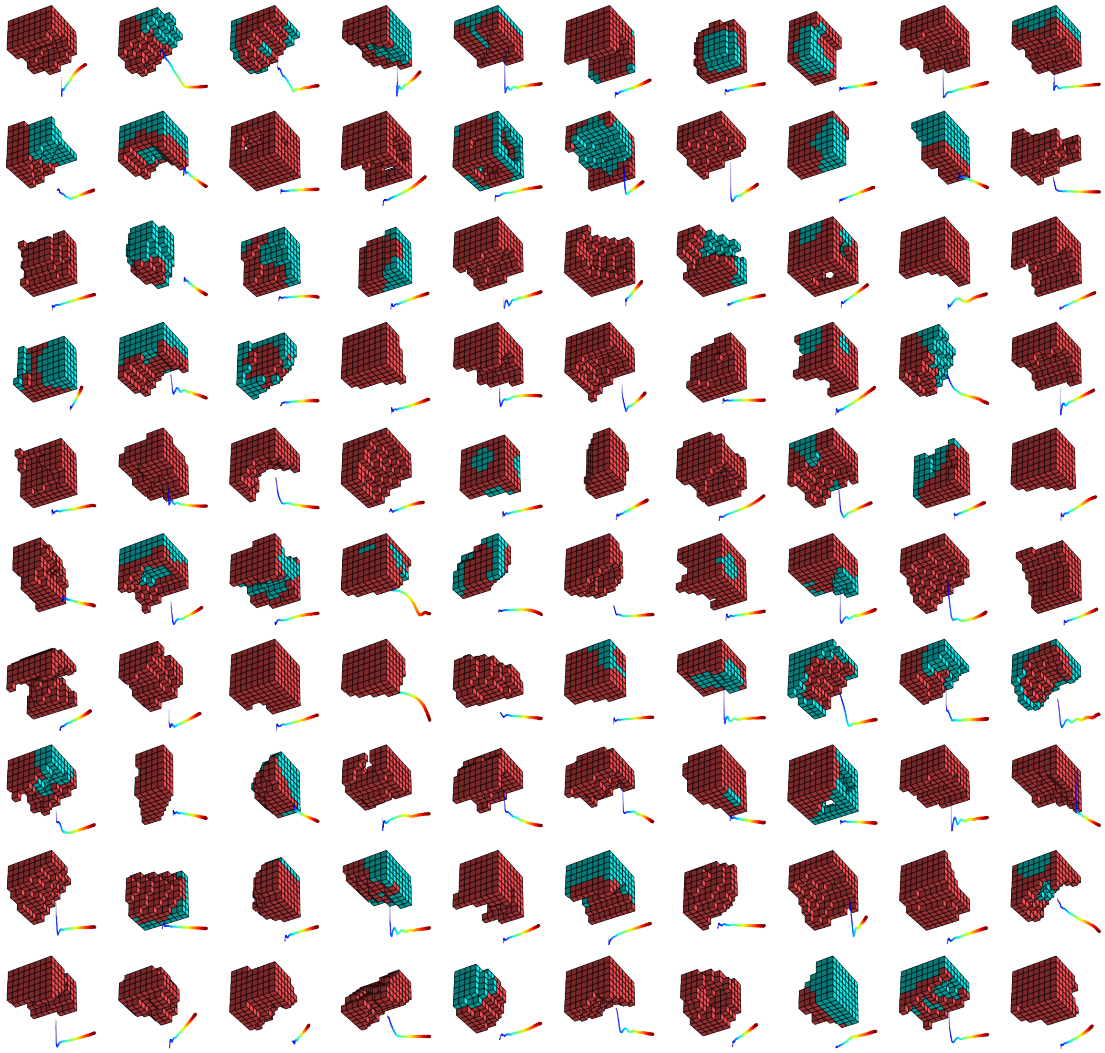


Figure S3. **The fastest designs from the second pass of the pipeline for locomotion.** The second pass of the pipeline for locomotion incorporated a few additional constraints: First-order hydrodynamics were added, tissue density was decreased (made lighter), tissue elasticity was decreased (made softer), and actuation was randomized (each voxel has a randomly-assigned phase-offset which is fixed in all descendants). Because actuation is provided by fewer and less-coordinated active cells, the behavioral trajectories (rainbow streaks below the design) have smaller amplitude, and the final displacement was decreased compared to the first pass (Fig. S2). Although designs were allowed to settle under gravity for one second before the evaluation period (and actuation) begins, an appreciable portion of the evaluated behavior sometimes began with the design settling to the floor (falling blue portions of the rainbow curves). (Behavioral streaks are scaled identically within this figure, but different from those in Fig. S2.) These additional constraints increased geometric diversity, promoted the use of passive (cyan) tissue, and generated behavior that better matched that of the biological representations.

A. Robustness filter.

Avg. velocity rank	Run	Failed conditional
1	90	B1, B3
2	75	B3
3	43	B2, B3
4	5	B3
5	48	B1, B2, B3
6	69	B3
7	51	B2, B3
8	88	B2, B3
9	20	B2, B3
10	76	B2, B3
11	79	B2, B3
12	12	B1, B2
13	84	B1, B2
14	29	B2, B3
15	**52**	**None**
16	99	B1
...	...	...
100	31	B1

B. Build filter.

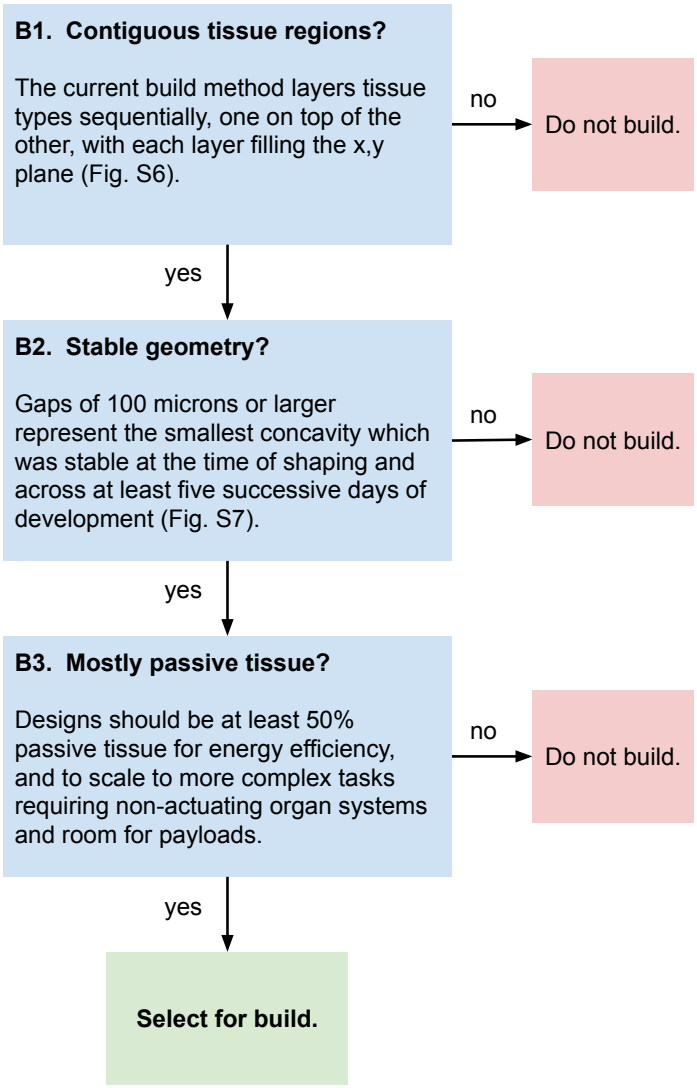


Figure S4. **The design filters and selection criteria.** The most performant design from each evolutionary run (the 100 designs in Fig. S3) are filtered in order to select the most promising one (in terms of estimated transferability, buildability, and scalability) for manufacture. The most performant designs are first ranked by their median performance (velocity) under random actuation (robustness filter; **A**). Then, in rank order, each design is evaluated by the three conditionals (**B1**-**B3**) in the build filter (**B**), which removes designs that are not suitable for the current build method, and/or those which are not scalable to more complex tasks in future deployments.

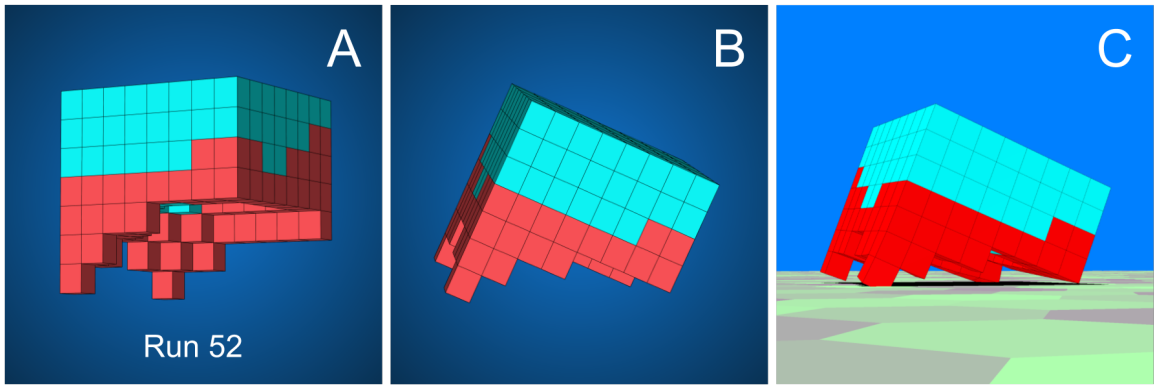


Figure S5. **The most robust design to pass through the build filter:** The champion of run 52 (locomotion; second pass) is drawn floating (**A**, **B**), and resting on the surface in simulation (**C**).

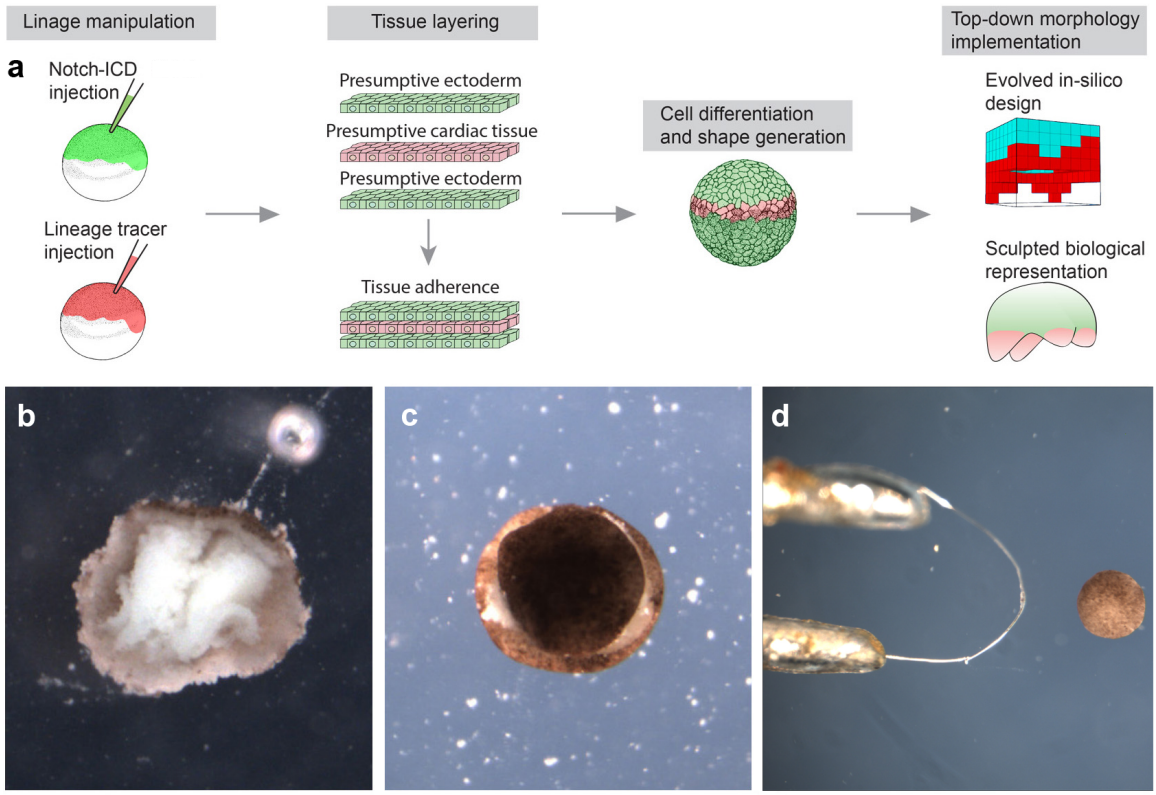


Figure S6. **Manufacturing cardiomyocyte-driven reconfigurable organisms.** mRNA constructs are delivered to cleave stage Xenopus embryos which inhibit multi-ciliated epithelial cell differentiation and enable tracking of heart muscle tissue, allowing multiple cell types to be combined and intelligently shaped (**A**). Tissues are layered sequentially, first with an underlying layer of unspecified epithelium upon which cardiac progenitor tissue is deposited (**B**). A second layer of unspecified epithelium is layered and allowed to heal for an hour (**C**). Shaping is then applied to contractile regions using a microcautery electrode to produce the final shape (**D**).

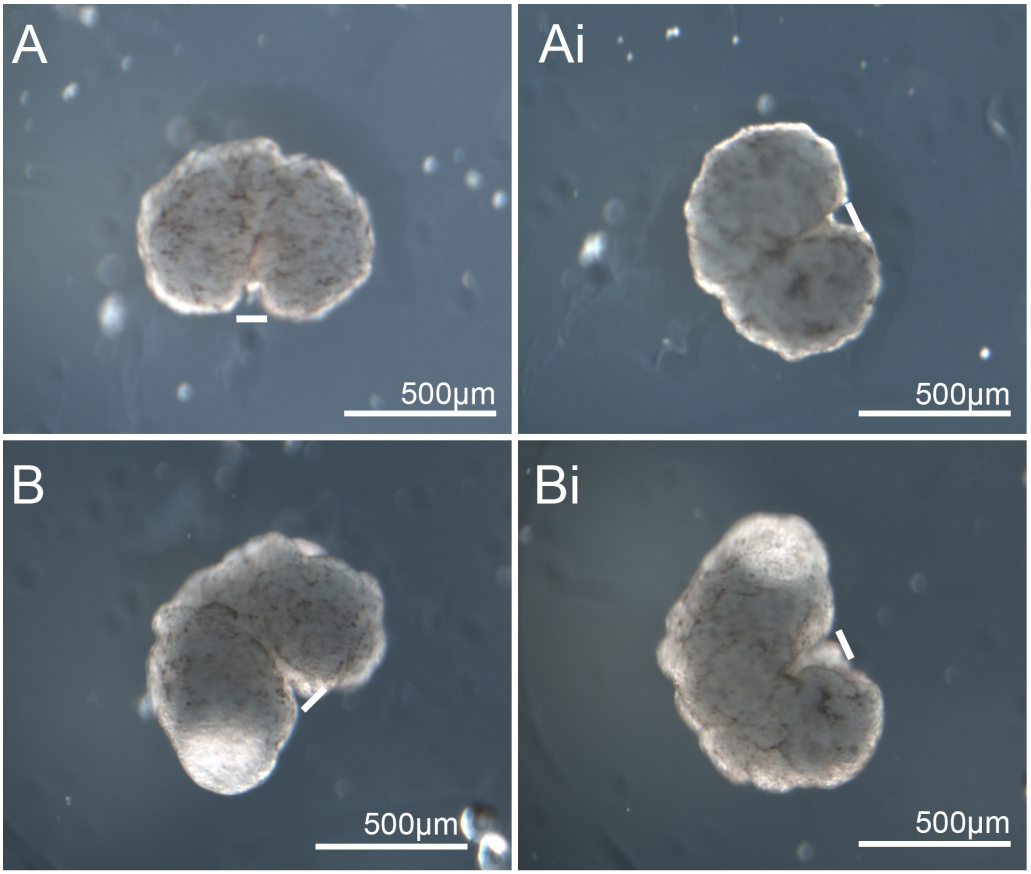


Figure S7. **Determining minimal concavity when shaping reconfigurable organisms.** To optimize the build filter when selecting in silico designs for biological sculpting, spherical biobots were subjected to progressively smaller shaping gaps to evaluate the minimal size which persists across the organisms lifespan. Gaps of 100 microns or larger (white line) represented the smallest concavity which was stable at the time of shaping (**A** and **Ai**) and across at least five successive days of development (**B** and **Bi**).

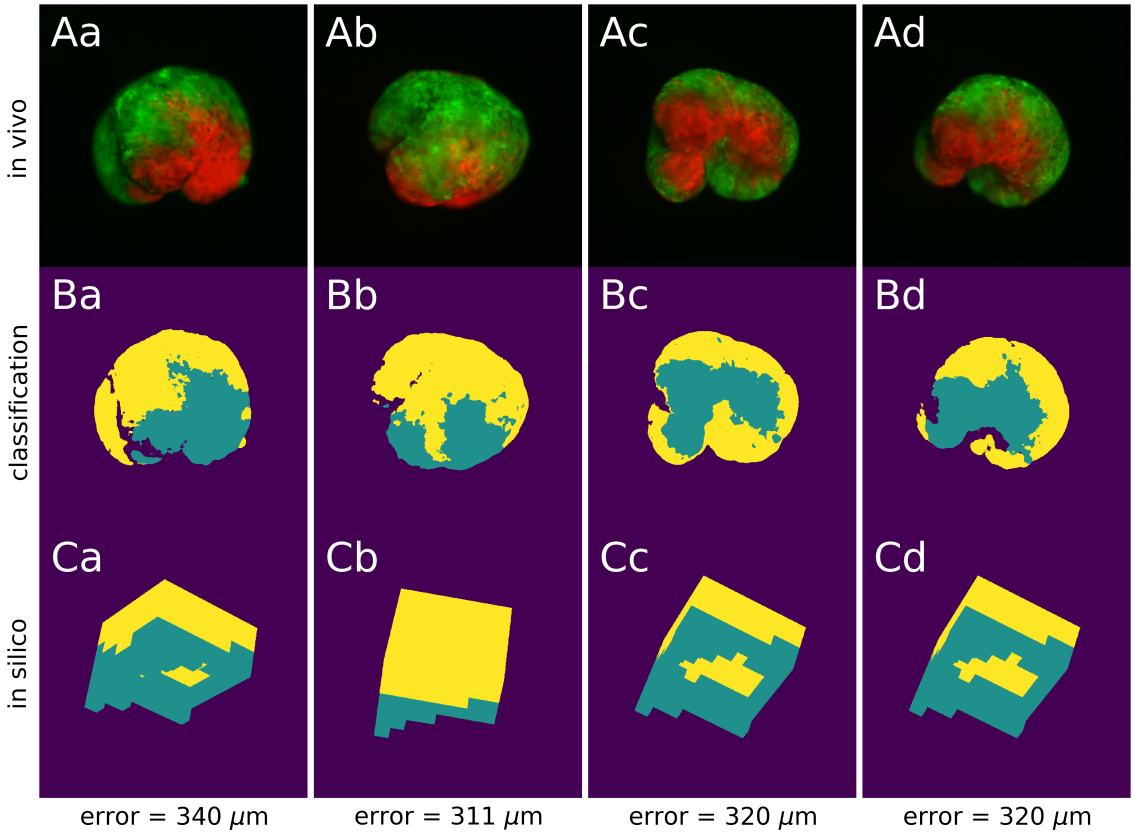


Figure S8. **Quantifying the structural match.** Lineage labeled organisms were imaged in multiple orientations (**A**) using a FITC filter set to visualize epidermal tissue (green) and a TRITC filter set to visualize cardiac progenitor (red) tissue. K-means clustering was used to classify each pixel as one of three tissue types: contractile, passive, or none (**B**). The Hausdorff distance between in vivo and in silico pixels (**C**) was calculated for each tissue type separately, in terms of euclidean distance in microns. Overall structural error is taken to be the larger of the two Hausdorff distances, computed for the two tissue types. The design is automatically cropped and rotated in 3D space to find the 2D perspective with the smallest error.

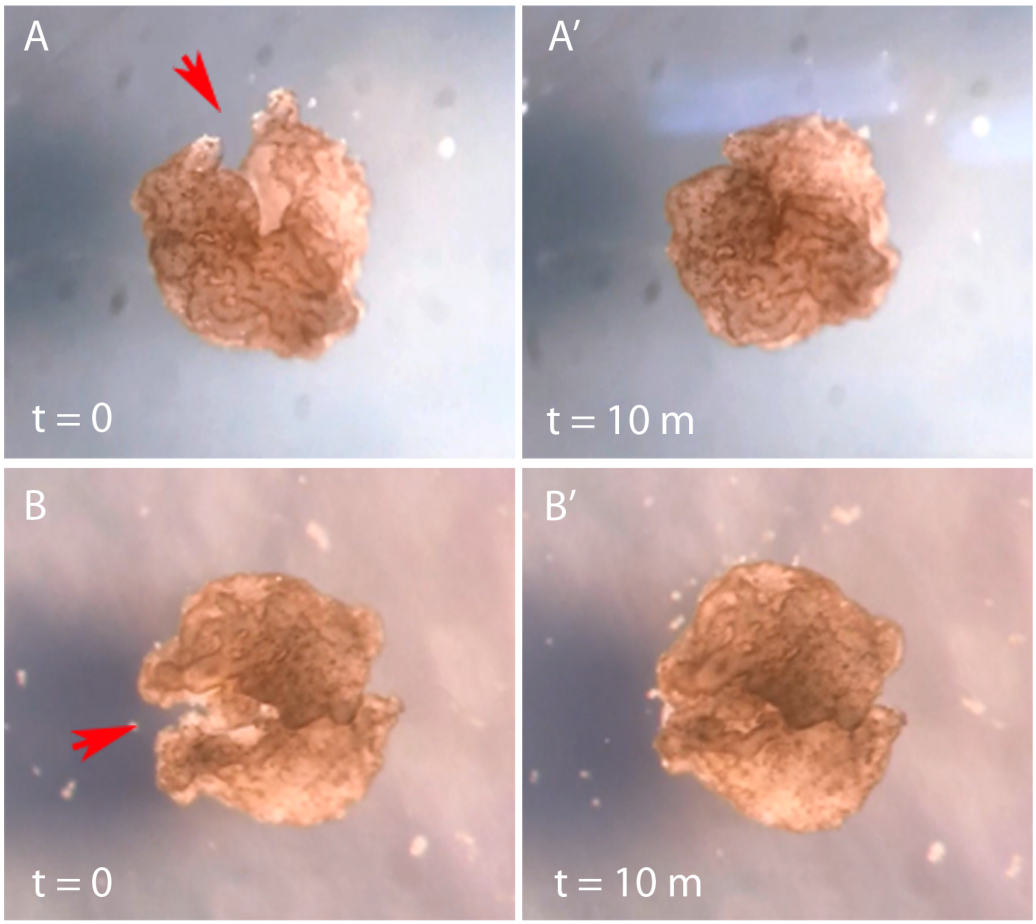


Figure S9. **Automatic self-repair in reconfigurable organisms.** One week-old non-ciliated reconfigurable organisms were subjected to mechanical laceration with microsurgery forceps (**A**, **B**). Each individual was filmed with time lapse imaging, every five seconds, for 10 minutes. In all cases, the biological representations were able to close the wound and survive through the remainder of the experiment (**A'**, **B'**). Red arrows indicate the site of mechanical damage. Organisms were able to heal much larger wounds during repair experiments than during their initial shaping (Fig. S7). This is because shaping requires subtraction of tissue over a longer period (which physically inhibits the wound from shutting) while repair involves a single laceration delivered in a short duration. Also, in repair experiments, the organisms possess a fully developed epidermis (which is expected to close wounds) compared to shaping experiments where all cells are embryonically derived and are still in the process of differentiation.

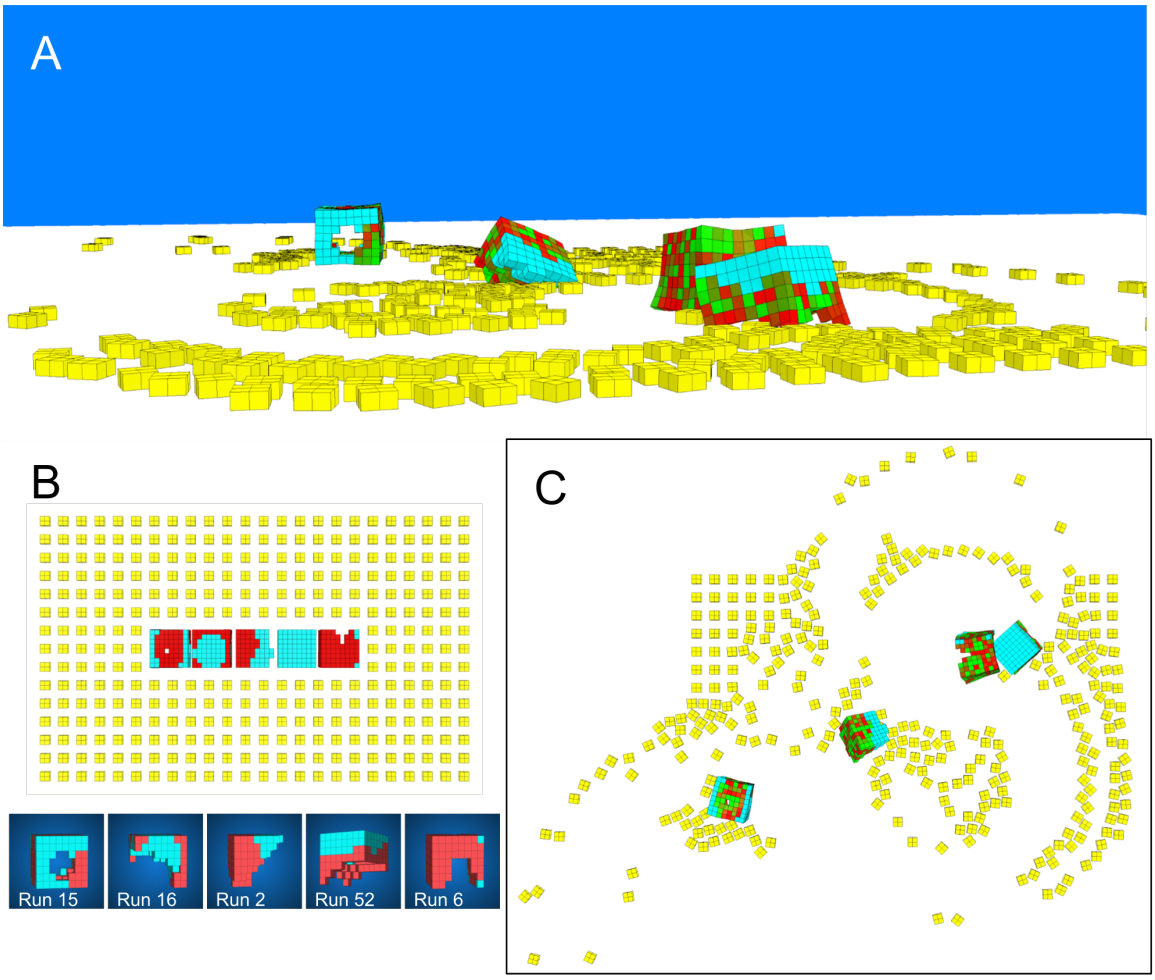


Figure S10. **Spontaneous collective behavior and particle aggregation in silico.** Collective behavior of multiple designs interacting with yellow particulate matter in the form of 2×2×1 voxel particles (**A**). Five designs (locomotion run champions, from the second pass; Fig. S3) are initially placed amid a 15×15 lattice of the debris (**B**). One of the designs (run 16) travels relatively straight, while the other four move in elliptical orbits, often colliding and coupling for multiple revolutions along a new orbit before collision with a third causes them to detach along a transposed version of their soloist trajectory (**C**). There was no top-down selection for such connections, they occur spontaneously and affect the manner in which the duo interacts with the external objects. In terms of clearing debris, the performance of the interlocked pair is sometimes more efficient than the sum of its parts.

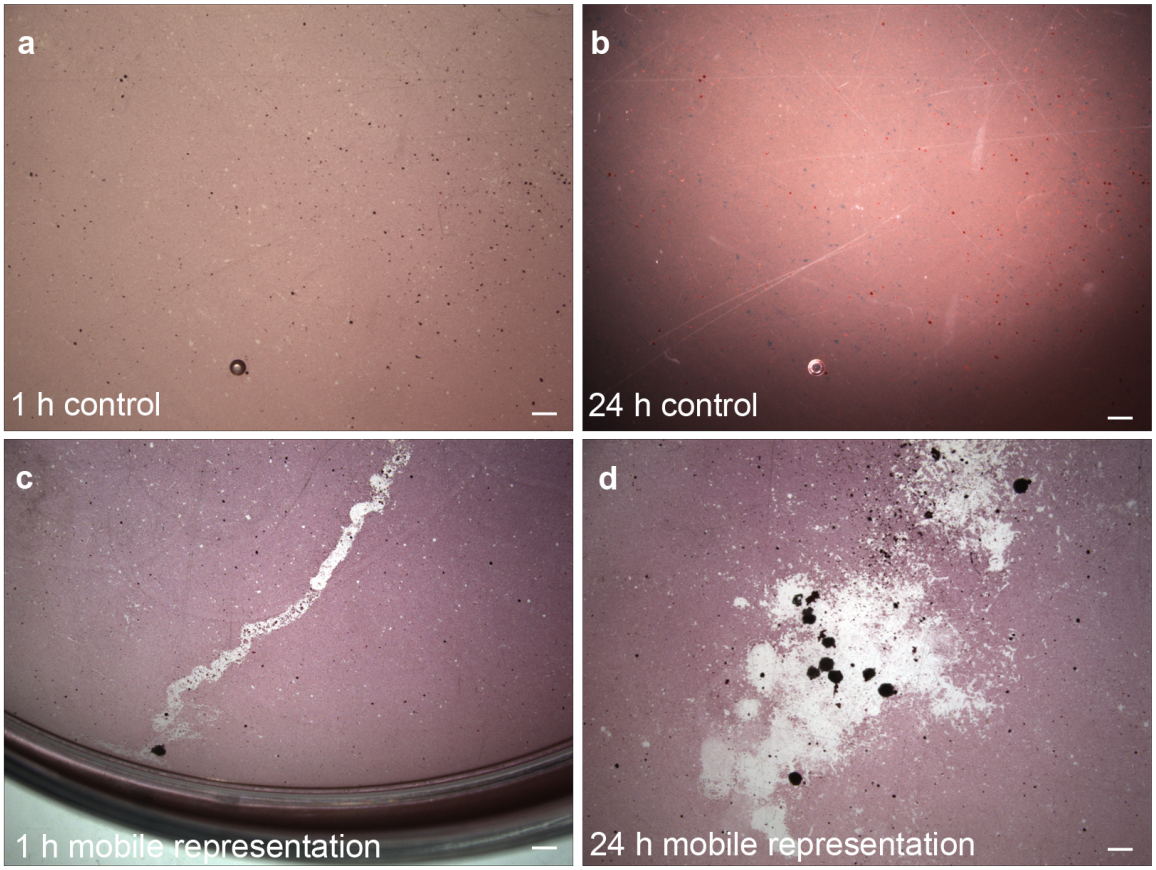


Figure S11. **Aggregation of carmine particles occurs only in the presence of organisms.** In the absence of reconfigurable organisms, carmine dye does not self aggregate after 1 or 24 hours (**a**, **b**). However, in the presence of self locomoting organisms, movement of dye particles by individuals can be observed after one hour (**c**), and collective aggregation is observed by 24h (**d**). Scale bar indicates 1 mm.

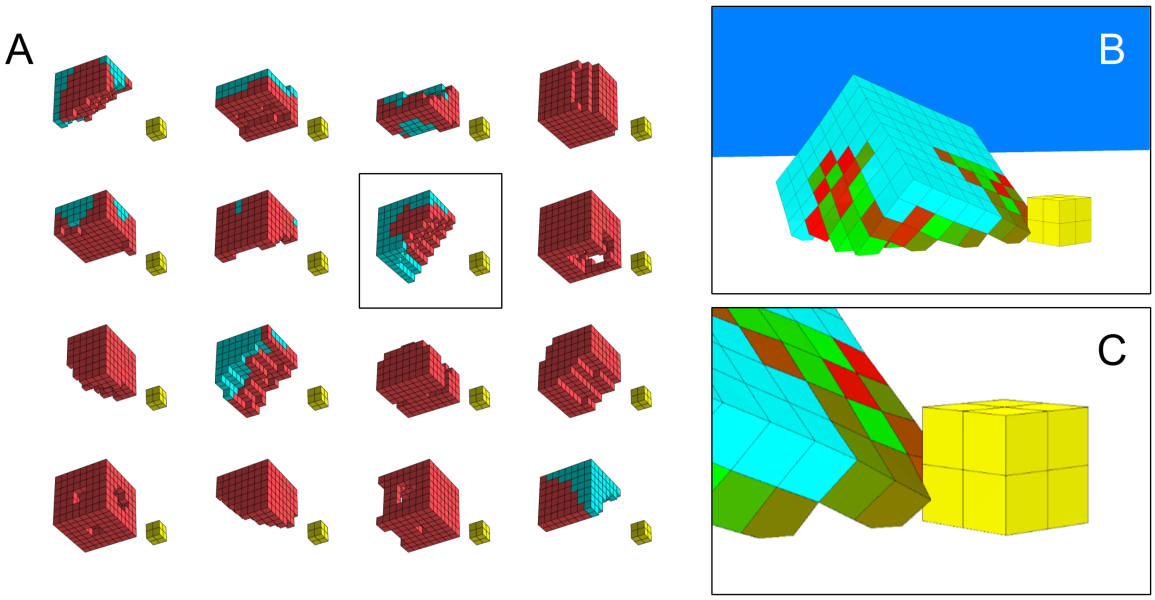


Figure S12. **Explicit selection for object manipulation in silico.** A new goal was input into the pipeline: maximize displacement of a 2×2×2 voxel object (yellow), during an evaluation period of 30 sec. Sixteen independent evolutionary runs (with random seeds 1-16) produced sixteen run champs (**A**). In some designs, a primitive end-effector—a notch in the corner of the body—evolved to hold and manipulate the object (**B**, **C**).

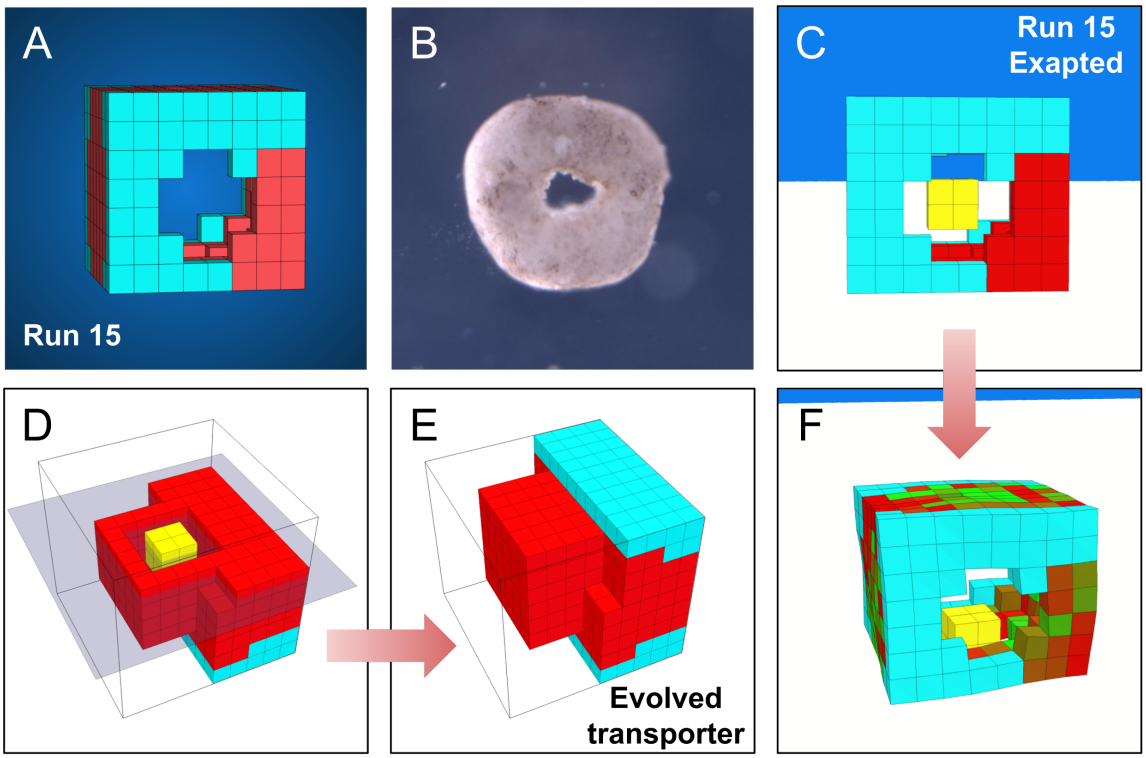


Figure S13. **From spontaneous to explicitly-optimized object transportation in silico.** Some designs evolved for displacement reduced hydrodynamic drag via a hole through the center of their transverse plane (e.g., the champion of run 15; **A**). This more complex topology was realized in vivo (**B**) but was not layered with contractile tissue. In simulation, this emergent feature was exapted as a pouch to store and transport objects (the 2×2×2 yellow "pills" in **C**). This realization led to a reformulation of the goal behavior for a subsequent round of evolution, in which pouches—with a free-floating pill inside them—were explicitly incorporated as a design constraint (cross-sectional view in **D**), and the new goal of maximizing the euclidean distance of the carried object was employed. This yielded evolved object transport in silico (**E**). Sixteen runs were performed with this new constraint and objective. The run 8 champion (seed=8) at the end of 350 generations is shown in D and E.

Figure S14. **Object expulsion in silico.** A non-locomotion-based goal was supplied to the pipeline: thrown object distance [40]. Using an additional objective that maximizes surface-area to volume ratio, limbs evolved which were used to throw a pink 2×2×2 object. For visibility, the voxels in this figure are colored by their angle relative to the ground plane, where green denotes parallel (angle zero), cooler colors (cyan and blue) denote voxel rotations into the right hand side (more or less) of the frame up to $-\pi$, and warmer colors (yellow, orange and red) denote voxel rotations into the left hand side up to $+\pi$. The run 2 champion (seed=2) at the end of 1000 generations is pictured. However, this design relies on finely-tuned actuation (which is unlikely to transfer) and fails to pass through the build filter because it is 100% muscle.

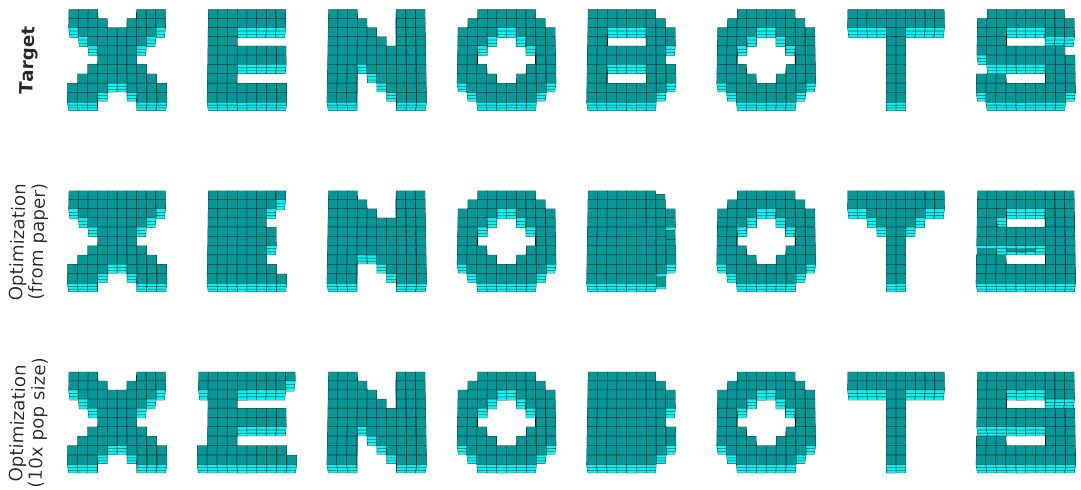


Figure S15. **The encoding bias.** Explicit geometry optimization (passive tissue only) within a 10-by-10-by-3 workspace. We chose seven unique target shapes—the voxelyzed letters "X", "E", "N", "O", "B", "T", and "S"—and employed the same indirect encoding (CPPNs; [221]), optimization algorithm (AFPO; [204]), and hyperparameters as in the main paper. Each shape can be represented as a bitstring with length 300, where each element corresponds to the presence (1) or absence (0) of a voxel at one of the 300 cartesian coordinates in the workspace. The objective function measures the Hamming distance between a given shape and the target. We re-optimized with a 10 times larger population size (500) and parallelized across 10 CPUs, which produced much more accurate solutions in the same wall-clock time (about 25 minutes per target). This suggests that shape (if not behavioral) complexity scales with population size. The difficulty of realizing certain target shapes with this encoding, is due to the particular activation functions we used—sin(), abs(), square(), sqrt(abs())—which tend to produce patterns that gradually vary in space, and thus bias search away from abrupt perimeter changes, such as the square cutaways and holes in "E", "B", and "S". This in-silico bias toward round edges mimics an in-vivo bias in our experiments: the pooled stem cells form a sphere during incubation, and naturally round-off abruptly carved edges after shaping. In future design-manufacture cycles, if particular structures are known to be obtainable in vivo and useful for a given task environment (e.g., long appendages for reaching), mathematical functions that promote such features (e.g., a step function) can be included in the in-silico design.

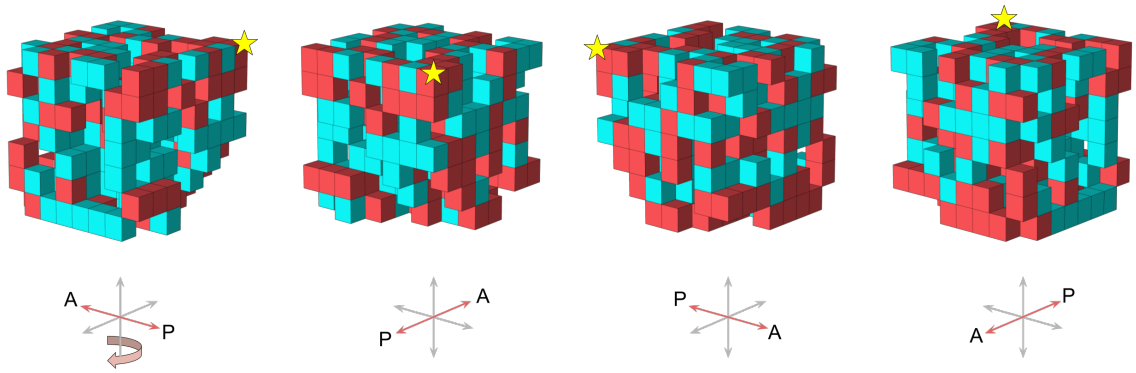
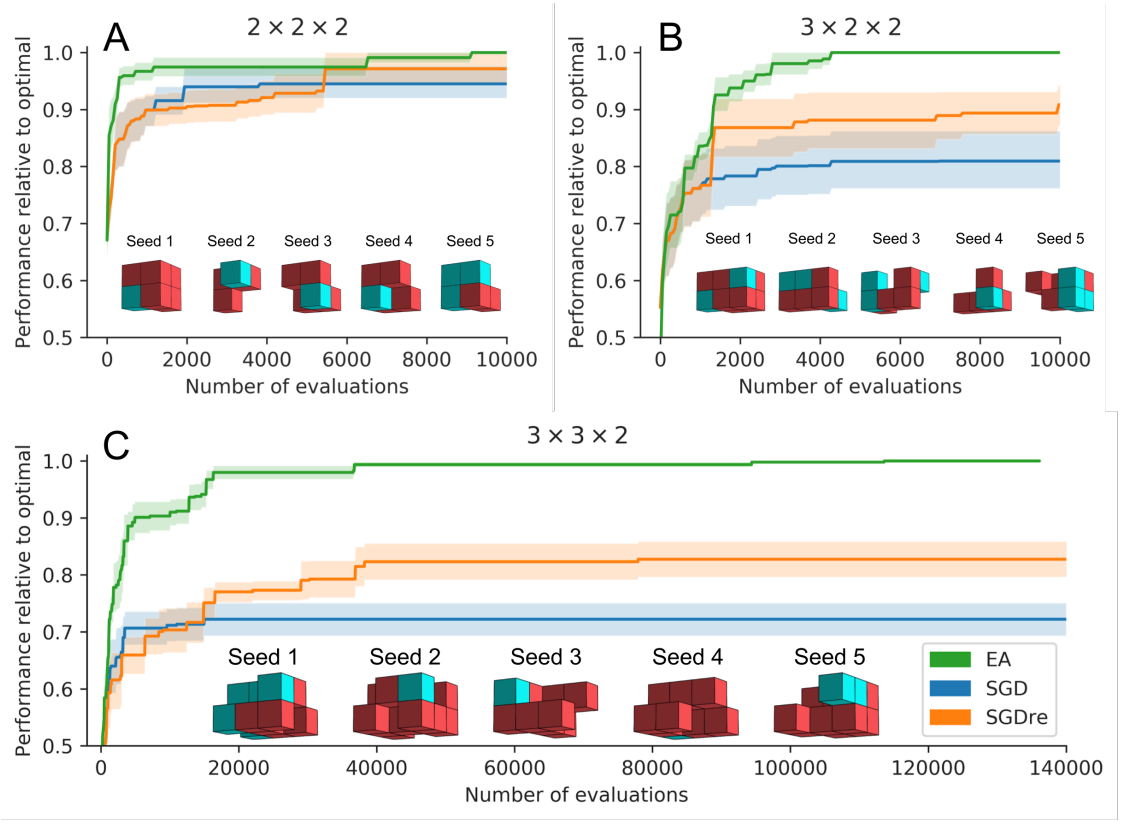


Figure S16. **Stochastic gradient descent directly applied to the design problem.** A policy gradient method [205] was paired with a popular SGD optimizer [112] and directly applied to the design problem given a random actuation pattern (seed=1). The solution returned by this algorithm is shown from four different perspectives, with a yellow star drawn above one of the voxels for reference. It is less performant (4.6 body lengths / min) than the solutions found by the evolutionary algorithm. But more importantly, it is not manufacturable using the current build method: it fails all three conditionals of the build filter (Fig. S4).



workspace	2x2x2	3x2x2	3x3x2	3x3x3	4x3x3	...	8x8x7
configurations	6,561	531,441	3.87×10^{8}	7.63×10^{12}	1.5×10^{17}	...	5.63×10^{213}

Figure S17. **Algorithm analysis.** The optimal design was identified in small solution spaces for five different random actuation patterns (seeds 1-5). The optimal designs in each workspace ($2\times2\times2$, $3\times2\times2$, and $3\times3\times2$) are pictured under their corresponding random seed. The evolutionary algorithm used in this paper (green curves; EA) is plotted alongside a policy gradient method [205] which used stochastic gradient descent (blue curves; SGD). Because SGD tends to prematurely converge to a suboptimal design, the algorithm was then modified to restart search from a new random initialization, every 1000 evaluations (orange curves; SGDre). The mean performance of each algorithm (relative to that of the optimal solution) is shown as a solid line. The shaded areas around the lines are bootstrapped 68% confidence intervals of the mean (which for normally distributed random variables, corresponds to plus/minus one standard deviation). The table at the bottom lists the total number of configurations in each workspace, not accounting for isomorphisms, such as translations or certain rotations, or those which result from taking the largest connected component to be the design.

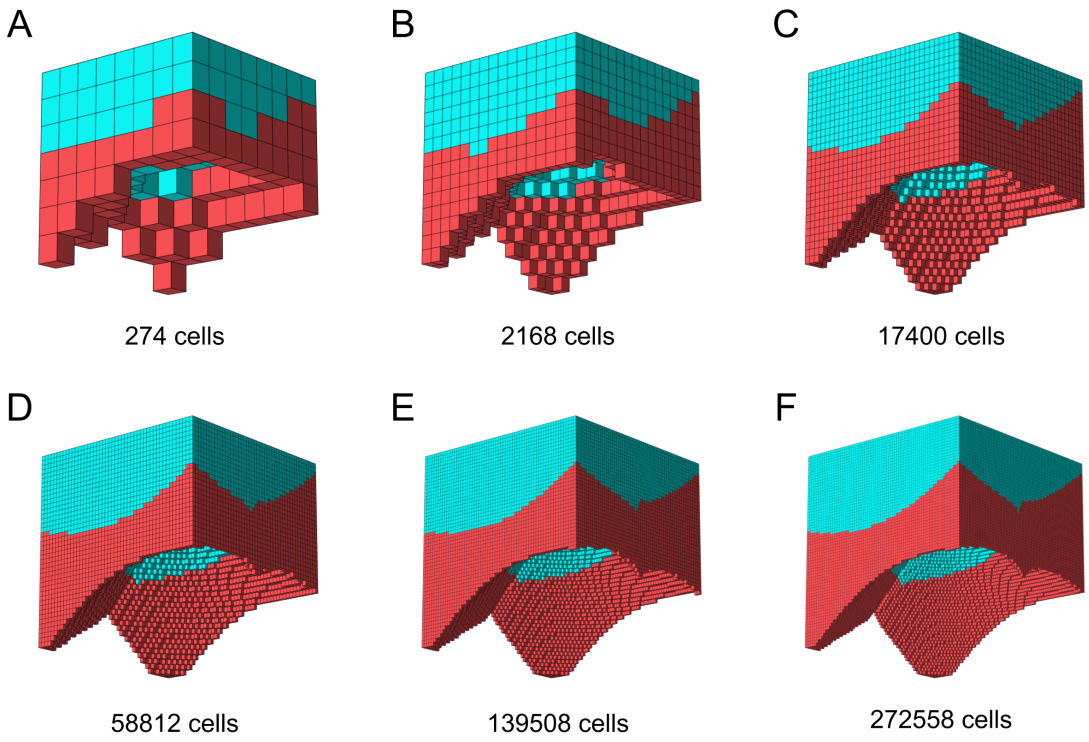


Figure S18. **Scaling the pipeline.** Designs were "carved" in silico from 8-by-8-by-7 workspace (448 voxel "cells") but the organisms were carved in vivo from a 10000 cell sphere, and their final geometries contain about 5000 cells. However, because the genetic encoding is scale-free, evolved designs (**A**) can be scaled to a higher resolution in silico (**B-F**) while preserving geometry, but not necessarily behavior. To determine the behavior of anatomically-upscaled designs, additional simulations are required, the CPU-time of which increases with each virtual voxel. Thus upscaling phenotypes can significantly slow the pace of evolution. However, a GPU-accelerated version of the simulator was recently released which, by updating voxels in parallel, may alleviate this concern.

Chapter 8

Argument

8.1 Précis of the Thesis

In Chapter 1, a history of evolved robots and protean machines situated the contributions of the present thesis in the literature. Two motivating observations were stated at the very beginning: (1) organisms are autonomous adaptive systems, and robots are not; (2) organisms are exceedingly protean systems, and robots are not. This thesis is at heart about closing the robot-organism gap by means of increasingly protean machines.

By incrementally incorporating morphological plasticity in the preceding chapters, it was possible to isolate how various aspects of development can affect and, under certain conditions, guide the evolution of adaptive behavior in embodied systems. Previously the evolution of development of artificial systems was investigated using either abstract nonembodied systems (e.g. bitstrings [96]) or software onboard morphologically-static hardware (e.g. neural network synaptic weights [77, 102]). Evolution and development were therefore confined to select subsets of actions among the fixed configuration space defined by the static body plan. Neither evolution or development could vary structure, shape or material properties to construct new configuration alternatives.

In our pursuit of increasingly protean machines, we are also interested in the interaction of subsystems unfolding at different timescales. Indeed, as machines become more embodied [113] and more protean, the number and kind of modalities that can vary and interact between timescales increases in exponential fashion. As a first step, we introduced a new mode of evolved development in artificial systems: shape change. By using embodied agents in a physically realistic environment, our model was able to strip away some of the assumptions required in previous work. Specifically, the evolution of open-loop "ballistic" shape change demonstrated that the explicit suppression of development enforced by Hinton and Nowlan [96], and those who built on their work, is in fact not necessary: ballistic development in physical systems is enough to increase evolvability (chapter 3).

Later in the thesis, shape was allowed to vary in addition to configuration patterns (chapter 4). Shape change occurred gradually and linearly from evolved infant to evolved adult forms. The temporal coordination of configuration oscillations was likewise shifted linearly across evolved initial and final phase-offsets. This allowed us to investigate how changes in shape and configuration-patterns might differentially affect the direction or rate of evolutionary search. More specifically, it exposed the previously unknown phenomenon of differential canalization reported here: Some initially developmentally plastic traits become integrated and canalized—if and only if they are robust to changes in other traits that remain plastic. This was missed in previous studies because they considered development of just a single modality: neural networks or abstractions thereof.

The robots presented here swept across a continuum of structures, shapes and material properties, as they moved through their environments. The intersection of these traits across different timescales generated positive and negative interference in terms of instantaneous velocity: robots sped up (synergy across developing traits) and slowed down (interference between traits) during various points in their lifetime. Unless all of the phenotypes swept over by an individual in development keep the robot motionless, there will be intervals of relatively superior and inferior performance. Evolution can thus improve overall fitness in a descendant by lengthening the time intervals containing superior phenotypes and reducing the intervals of inferior phenotypes. However, this is only possible if such mutations exist. We have found here that such mutations do exist in cases where evolutionary changes to one trait do not disrupt the successful behavior contributed by other traits.

The finding of differential canalization has important implications for protean machines since it introduces a form of developmental modularity that allows evolution to improve one subsystem without disrupting others. This might help explain why organisms have any persistent canalized forms at all. It also provides a mechanism through which exceedingly protean machines could settle into a recognizable shape while operating in a relatively static environment. A temporally-clamped protean machine might even look like a conventional robot for an extended period of time, but unlike paradigmatic robots, when the environmental conditions change, a protean machine is not bounded by a static body plan with fixed sensorimotor constraints and well-specified configuration spaces.

The current paradigm of robotics relies on well-specified components that are designed to physically interact in particular ways with each other and the outside world. A sequence of steadily more protean machines, in contrast, will have by definition progressively ill-specified body plans. This allows increasingly protean machines to gradually distance themselves from their designer. This is a fundamentally different form of autonomy than a robot can achieve with a well-specified static morphology. Changing a morphologically-static robot's self image [27, 55], for instance, only

changes the way the robot maps sensor data to the sequences of configurations it will execute. In addition to finding an appropriate mapping from sensors (or central pattern generators) to configuration, there is also the problem of determining the basic categories of configurations: the configuration space.

By constructing their own structures, shapes and material properties, the protean machines in this thesis created their own means of interacting with their environments. Some formed spheres (chapter 4); others grew long limbs (chapter 6); some grew analogues of bones and calluses; and yet others grew stronger or additional motors (chapter 5). In each case there was functional emergence: the same sequence of actions led to different configurations, different ways the robot pushed and pulled against its environment to propel itself forward or influence objects in its environment (such as pellets or other agents; chapter 7).

This thesis focused on soft robots, but if a rigid-bodied robot can contort and reorient its well-specified jointed collection of rigid links into a new resting shape, and vary its configuration about that new shape, then it is likewise, by definition, a protean machine. Such a machine could construct its own means of influencing the world in the same way insects can rub their wings or legs together to generate sounds. However, a robot with n rigid links is also by definition less protean than a robot with n^2 rigid links, or one that can change the rigidity, volume, number and placement of its links.

This thesis suggests that autonomy is a spectrum that by and large aligns with the breadth and depth of morphological plasticity. A developing yet morphologically-static robot [27, 28, 55, 77, 102] can intelligently select sequences of configurations among those available given its current body plan, but it cannot deform to regenerate configuration space lost to structural injury, nor can it morph to expand the set of configurations made available to it externally from the outset (deployment/birth). It follows that a more protean machine can potentially be more robust, adaptive and autonomous than an otherwise equivalent but less protean machine.

Designing protean systems, however, is both operationally and conceptually challenging. Perhaps because of the former reason, despite early investigations by Pask and Beer (and a mini-resurgence at Sussex University in the 1990s), protean machines remain a neglected research program. To overcome some of the conceptual challenges that arise when designing protean machines, this thesis demonstrated a variety of automated design tools built up over the past 25 years in the field of evolutionary robotics, and added a few new ones along the way. To overcome some of the operational concerns that emerge when building robots from artificial materials, an algorithmic pipeline was introduced for the automated design of completely biological machines, which are popularly referred to as "xenobots".

Prior to the work presented here, very few cases of evolved physical robots had been reported in the literature [33, 38, 70, 94, 99, 182]. In each instance, structure

and configuration patterns were evolved, but the robot could not change its body plan during operation. The first experiment reported here likewise evolved ontogenetically static structures which were manufactured in reality (chapter 2). To demonstrate the value of incorporating ontogenetic morphological plasticity, the next set of experiments considered the evolution of ballistic ontogenetic shape change in silico (chapter 3). Plasticity was then marginally increased by evolving ballistic ontogenetic change not only to resting shape but also to the coordination of configuration oscillations occurring on a faster timescale in silico (chapter 4). The feedback loop between environment and development was then closed by the evolution of ontogenetic changes in material properties in response to environmental signals in silico (chapter 5). In the next set of experiments, configuration oscillations were evolved for a fixed structure, which was later exposed to a series of increasingly deeper structural amputations; for each damage scenario, the robot automatically deformed its shape so as to recover function without changing the evolved configuration oscillations (chapter 6). Finally, we saw the design and manufacture of novel organisms, which represent an alternative path to protean machines, one in which there is no robot-organism gap, but does raise its own unique set of engineering challenges (chapter 7).

The following sections summarize each of these chapters in turn.

8.2 Sim-to-Real for Structure

In Chapter 2, a low cost, open source, and modular soft robot design and construction kit was introduced and used to transfer an order of magnitude more robot designs from simulation to reality than any other method to date.

Protean Capacity

- *Phylogenic change*: structure, material distribution.
- *Ontogenic change*: configuration.

Realization

- *Material*: passive and pneumatically-actuated silicone voxels (cubic bladders).
- *Structure*: number, placement and kind of voxels.
- *Shape* [fixed]: atmospheric pressure (cubes).
- *Configuration*: pressure oscillation of all active voxels in unison (expansion, compression back to cube, hold at cube, repeat).

8.2.1 Contribution

Over the past two and a half decades since Jakobi's original sim2real experiments [108], much work has sought to regularize and tune computational models of robots so that control policies optimized in simulation are just as successful when tested on the physical system [3, 27, 41, 55, 84, 104, 116, 127, 166, 230, 236]. That is, past work has extensively studied the simulation-reality gap for configuration controllers. With the exception of Rosser et al. [197], who transferred parts of a robot (flapping wing designs), little to no work has investigated the simulation-reality gap for whole robots as a function of their morphology. By transferring a diversity of locomoting soft robot designs from simulation to reality, the reality gap was measured as a function of robot structure given a static control policy. Under one measure (net displacement) the reality gap appeared rather small, but under another (velocity) the gap was much wider. This illuminated the fact that there are reality *gaps* rather than a single gap. This suggests that, if structure is allowed to vary, automated design could be used as a filter to discover body plans with the smallest gaps, before utilizing modern sim2real protocols for controllers.

The kit's affordability, safety, speed, and simplicity effectively lowers the barrier of entry to soft robotics for non-experts. Such a tool is poised to generate increasingly more, and more reproducible, data about the design of protean machines. In the age of Covid-19, a do-it-yourself-at-home soft robot kit is particularly appealing and has thus far been adopted by six soft robotics research labs spanning four countries (Denmark, Japan, UK, USA).

8.3 Ballistic Development

In Chapter 3, a minimal yet embodied model of development was introduced in order to isolate the effect of ontogenetic morphological change, without the confounding effects of environmental mediation: The shape of the robot changes over its lifetime, yet development is not influenced by the environment. We refer to this as *ballistic development*.

Protean Capacity

- *Phylogenic change*: shape trajectory.
- *Ontogenic change*: shape, configuration.

Concretization[1]

- *Material* [fixed]: actuated voxels (1 to 6 elastic beams).
- *Structure* [fixed]: a 4-by-4-by-3 block of voxels (48 voxels; 108 elastic beams).
- *Shape*: evolved initial and final sets of rest volumes (resting beam lengths) for each voxel; linear ontogenetic shape change between sets.
- *Configuration* [fixed]: voxel length oscillation, all voxels in unison (expansion/-contraction according to sine wave with frequency 4 Hz).

8.3.1 Contribution

It was shown that even this simple developmental model confers evolvability because it allows evolution to sweep over a larger range of body plans than an equivalent non-developmental system, and subsequent heterochronic mutations "lock in" this body plan in more morphologically-static descendants.

Ontogenetic shape change naturally provided a continuum in terms of the magnitude of mutational phenotypic impact, from the very large (caused by early-in-life developmental mutations) to the very small (caused by late-in-life mutations). It was predicted that, because of this, such a developmental system will be more evolvable than an equivalent non-developmental system because the latter lacks this inherent spectrum in the magnitude of mutational impacts.

8.4 Differential Canalization

In Chapter 4, a previously unknown phenomenon was discovered when simulated protean machines were allowed to develop and evolve: Evolution discovers body plans robust to control changes, these body plans become genetically assimilated, yet controllers for these agents are not assimilated. This allows evolution to continue climbing fitness gradients by tinkering with the developmental programs for controllers within these permissive body plans. This exposes a previously unknown detail about the Baldwin effect: instead of all useful traits becoming genetically assimilated, only traits that render the agent robust to changes in other traits become assimilated. We refer to this as *differential canalization*.

[1]This particular capacity for morphological change was concretely demonstrated in a computational yet embodied model (in silico), however it was not *realized*: no physical robot was used during hypothesis testing.

Protean Capacity

- *Phylogenic change*: shape trajectory, configuration trajectory.
- *Ontogenic change*: shape, configuration.

Concretization

- *Material* [fixed]: same as chapter 3.
- *Structure* [fixed]: same as chapter 3.
- *Shape*: same as chapter 3.
- *Configuration*: voxel length oscillation phase shifted from a central pattern generator (sine wave with frequency 4 Hz). Evolved initial and final of phase offsets for each voxel; linear ontogenetic phase-offset change between sets.

8.4.1 Contribution

The key observation here is that only phenotypic traits that render the agent robust to changes in other traits become assimilated, a phenomenon we term differential canalization. This insight was exposed by modeling the development of simulated robots as they interacted with a physically realistic environment.

In 1987, Hinton and Nowlan [96] showed how the Baldwin effect could speed evolution in the presence of a "needle in the haystack": an abstract trait that is only of value when perfect. In chapter 4, an embodied needle in the haystack was found: a rolling ball. On flat terrain, rolling can be much faster and more efficient than walking, but finding such a design is difficult in morphologically-static robots because (1) rolling over once is much less likely to occur in a random individual than shuffling forwards slightly; and (2) rolling just once incurs the fitness penalty of having fallen over and thus not being able to subsequently walk for the rest of the trial.

Among the oscillating shapes swept over by the evolution of ballistic development, some had limbs and shuffled, while others had smoother curvatures and rocked back and forth. In the lineages of the most fit individuals, rolling was initially discovered by a distant ancestor at the very end of their ontogeny. A form of progenesis was then observed in which heterochronic mutations generated descendants with reduced morphological plasticity in each voxel. This gradually preponed rolling from a late onset behavior to one that arose increasing earlier in ontogeny.

Shape was thus canalized in the rollers, but configuration patterns were not. This is because a spherical body shape is a "permissive body plan". Simply put, practically all oscillation patterns in an asymmetrical ball will result in locomotion. In fact,

thousands of random mutations to the ball did not significantly affect its performance. Legged body shapes, in contrast, were not permissive: they were finely tuned to match their configuration oscillations (and vice versa). A sequence of four or five random mutations would always break the functionality of a legged walker. Because mutations to legged bodies needed to be precisely coordinated with mutations in configuration patterns, highly fit legged morphologies could not be canalized.

These results demonstrate how incorporating morphological development in the optimization of robots can reveal, through differential canalization, characters which are robust to internal changes. Robots that are robust to internal changes in their controllers may also be robust to external changes in their environment [28]. Thus, allowing robots to change their structure as they behave might facilitate evolutionary improvement of their descendants, even if these robots will be deployed with static phenotypes or in relatively unchanging environments.

8.5 Environment-Mediated Development

In Chapter 5, simulated robots modified their own material properties in response to environmental conditions, according to evolved "laws" of ontogenetic change akin to Wolff's law of bone growth [252] or the soft tissue corollary, Davis' law. By doing so we realized robots that were equally fit but more robust to extreme material defects than robots that did not develop during their lifetimes, or developed in response to a different interoceptive stimulus.

Protean Capacity

- *Phylogenic change*: structure; initial material distribution and gain (rate of change given stimuli); configuration trajectory.
- *Ontogenic change*: material, configuration.

Concretization

- *Material*: dense voxels with evolved initial elasticity (Young's modulus between 10^4 and 10^{10} MPa), and evolved rate of ontogenetic softening/stiffening in response to engineering stress / pressure.
- *Structure*: evolved number and placement of voxels within a 10-by-10-by-10 voxel workspace.
- *Shape* [fixed]: cubes.

- *Configuration*: voxel length oscillation phase shifted relative to a central pattern generator with a phase offset of zero (sine wave with frequency 5 Hz).

8.5.1 Contribution

This work demonstrated that robustness can be achieved in artificial systems by evolving the structure of robots, their configuration patterns, and how their material properties develop in response to particular interoceptive stimuli during their lifetimes. Because development was manually removed before testing, robustness was not a matter of changing one's body, as in the example of plant growth [226]; rather, it was an intrinsic property (a genetic bias) of the evolved structure and configuration patterns.

We found that different types of developmental feedback loops elicited different evolved properties. For instance, if one modality (stiffness) responds to one particular internal signal (engineering stress) but not another (pressure), robots evolved structures that intrinsically buffered large deviations from their expected material properties. In other words, morphological change in response to the appropriate signals during evolution can foster "zero-shot" generalization. Meanwhile, developmental reactions in response to pressure increased evolutionary divergence: pressure-adaptive robots evolved more diverse structures than stress-adaptive robots. This suggests there may be other developmental feedback loops that could be made available to evolution that would lead to more diverse and robust robots.

8.6 Shapeshifting for Damage Recovery

In Chapter 6, for the first time, a robot automatically recovered from unanticipated damage by deforming its resting shape without changing its control policy.

Protean Capacity

- *Phylogenic change*: configuration trajectory.
- *Slow ontogenic change* [damage]: structure.
- *Medium ontogenic change*: shape.
- *Fast ontogenic change*: configuration.

Realization

- *Material* [fixed]: pneumatically-actuated, hollow silicone voxels.

- *Structure*: various amputations from quadrupedal form.
- *Shape*: learned resting pressure for each damage case.
- *Configuration*: pressure oscillation phase shifted from a central pattern generator (sine wave with frequency 5 Hz).

8.6.1 Contribution

A new approach to robot damage recovery was proposed. Instead of treating the body as just the problem domain [27, 55], we here modified it as part of the computational loop. That is, instead of presenting the remnant shape of the damaged robot to optimization as fixed, we enabled optimization to fundamentally deform this shape as the essential part of the recovery process. In doing so we realized a machine that recovered more function than an otherwise equivalent system that could adapt its controller but not deform its shape.

Because the robot had ill-specified shape, it could deform to construct its new effectors, surface contact geometries, and novel mechanisms for the storage/release of elastic strain energy. Using a simple evolutionary algorithm, the robot automatically generated and tested body shapes until it found shapes that maintained forward locomotion in the face of unexpected structural changes.

We found that, especially in the case of "deep insult", such as removal of all four of the robot's legs, the damaged machine evolves shape changes that not only recover the original level of function (locomotion) as before, but can in fact surpass the original level of performance (speed). This suggests that shape change, instead of control readaptation, may be a better method to recover function after damage in some cases.

8.7 Computer-Designed Organisms

In Chapter 7, I described the first end-to-end automated design of novel organisms: AI methods design unique organisms in simulation, after which they are rapidly realized as living systems using a cell-based construction kit. This yields a continuous flow of synthetic living organisms or "biobots" that perform useful tasks yet bear little resemblance, above the cellular level, to any existing organisms.

Protean Capacity

- *Phylogenic change*: structure, material; configuration variance.
- *Slow ontogenic change* [in vivo]: structure, shape, material.

- *Fast ontogenic change*: configuration.

Realization

- *Material*: initially pluripotent stem cells, later fated to become specific epidermal and cardiomyocyte cell lineages.
- *Structure*: evolved placement and distribution of cell types.
- *Shape*: cells compact together and conspire to self-heal lacerations in their collective structure (mechanism currently unknown; it's possible the tissue simply zippers shut through the action of cadherins, gap junctions, and tight junctions on neighboring cells).
- *Configuration*: contraction/relaxation of cardiomyocytes (temporal coordination in novel structures is currently unknown).

8.7.1 Contribution

Growing interest in AI has to date been restricted to technological constructs such as neural networks, robots, or autonomous cars. The experiments here introduced the first artificial, intelligent, yet fully biological constructs. While automatically designing machines in silico and manufacturing them as robots using 3D printers is now well established, automatically designing and instantiating living systems has never before been demonstrated. Recent efforts to build novel living machines [56, 72, 93, 169, 176, 192, 195, 253] used designs created by the investigators, rather than AI-generated ones. Although the fields of synthetic biology and organoid design have made inroads into designing living systems, the former is restricted to genetic modification of existing organisms while the latter is focused on designing only parts of a whole organism. This work, in contrast, describes a novel path to creating synthetic organisms out of cells without genomic editing.

Chapter 7 demonstrated how to evolve novel organisms for specific behaviors entirely in simulation and then build them by combining together different biological tissues. This is possible despite the fact that we do not fully understand how cells communicate and cooperate to build and maintain functional bodies. Specifically, it is currently unknown how cardiomyocytes will synchronize their contractions when reorganized into novel structures in vivo. In order to avoid injecting incorrect assumptions into the simulations, cardiomyocyte temporal coordination was modeled as random noise. As a side effect of selection pressure for locomotion, derandomizing morphologies evolved: evolutionary improvement occurred through changes in overall structure, and distribution of the passive and contractile cells, to collectively

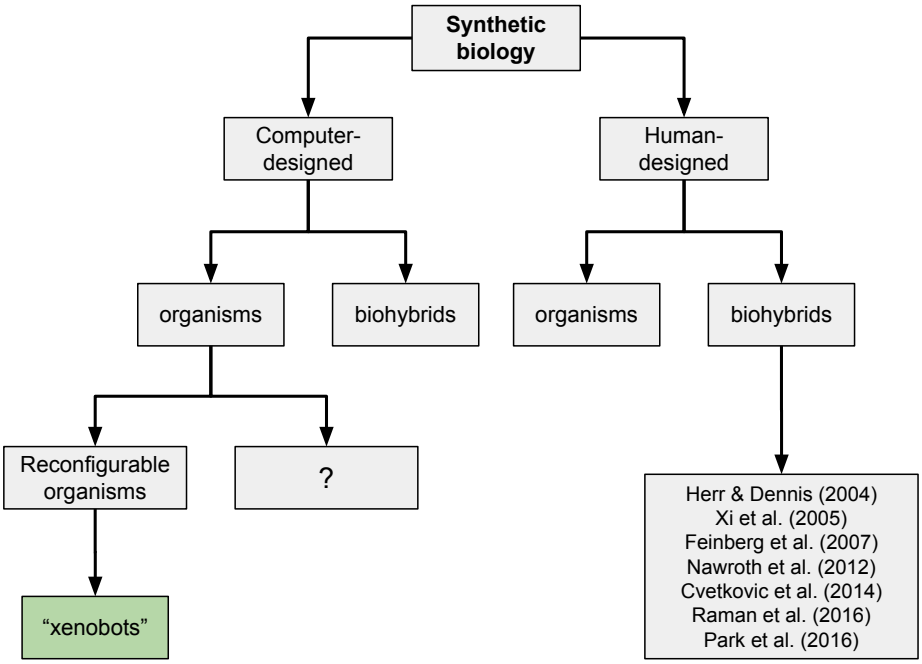


Figure 8.1: **Taxonomy of synthetic biology.** Xenobots are a particular species of Reconfigurable Organisms, which are an instantiation of Computer-Designed Organisms, a mode of Synthetic Biology distinguished from human-designed biological constructs, which includes all organiods and biohybrids built to date.

derandomize the global movement produced by the random actuation. The resulting organisms therefore embodied not only the structure of evolved in silico designs but also their behavior. Thus, by virtue of structure and distribution of material (tissue types), new kinds of domesticated organisms can be built and programmed to perform custom tasks.

8.7.2 Future Work

The first computer-designed organisms were called reconfigurable organisms (Fig. 8.1) because their configuration spaces were changed by combining together different cells into novel structures. This is a reference to reconfigurable modular robots in which robot modules are free to attach/detach during assembly and sometimes behavior, thus blurring the line between structure and configuration [180]. Xenobots are a particular kind of reconfigurable organism. If we made a batch of reconfigurable organisms out of cells taken from early embryos of the African elephant (*Loxodonta*) instead of the African clawed frog (*Xenopus laevis*) then they would be called "loxobots" not "xenobots". There is also the possibility of chimeras (mixtures of species) and

biohybrids (mixtures of living and artificial materials).

Reconfigurable organisms are assembled according to a blueprint which could be computer-generated or human-generated. The blueprint tells a microsurgeon or 3D bioprinter, at some resolution of detail, precisely where all the tissues should go in relation to each other. But there are other ways computers could design bespoke organisms. Rather than a blueprint of a specific target form, a computer-designed recipe could indicate how and when to apply various biochemical/biomechanical/bioelectrical manipulations so as to deflect development toward a target form and/or function. With constant feedback from silico to vivo and back, the pipeline would learn to influence the cells' self-assembly instead of imposing sometimes unbuildable constraints on their structure. These self-reconfigurable computer-designed organisms would thus obviate the need for the build filter described in chapter 7, and could in theory produce any structure or organ seen in nature (hands, eyes, noses, brains) and others yet to be discovered by evolution on Earth.

Despite our initial demonstrations that such is possible, it is still unclear which computer-designed organisms are easy, difficult or impossible to bioengineer in reality. Computer-designed organisms are, in many ways, the inverse of traditional robots. Many behaviors that are difficult to instantiate in robots composed of metal and electronics such as self assembly, -reconfiguration, -reproduction, and -repair, are ubiquitous in living materials. On the other hand, we can fabricate just about any reasonably scaled and shaped structure using steel, concrete, or plastics; living systems tend to resist adopting new forms imposed on them. The xenobots, as just one example, tend to smooth sharp edges and close small concavities in their sculpted body plans during development.

Collaboration across traditional academic departments (not just between computer science and developmental biology, but also mechanical-, chemical-, biomedical-, and molecular engineering) will be necessary to more adequately investigate and exploit this strange inversion of engineering difficulty.

8.8 Conclusion

This thesis presented six sets of experiments (chapters 2 to 7), in which the morphologies of robots were permitted to vary on increasingly more scales in space and time. The results suggest that the evolution of development can fortify autonomous machines against entropy, and expedite the acquisition of performant robust behaviors, even if the system is later deployed with a relatively static morphology. The penultimate chapter described the first organisms to evolve in computer simulations, instead of physically on Earth. By building novel multicellular organisms without genetic engineering, we have learned a great deal about the phenotypic plasticity inherent

in development but missing in conventional robots. Ultimately, this knowledge will be put to work developing useful protean technologies composed of living systems, artificial materials, or admixtures of both.

Bibliography

[1] The Online Encyclopedia of Integer Sequences (Nos. A000162 and A001931). https://oeis.org/A000162.

[2] David Ackley and Michael Littman. Interactions Between Learning and Evolution. *Artificial Life II*, 1992.

[3] Ilge Akkaya, Marcin Andrychowicz, Maciek Chociej, Mateusz Litwin, Bob McGrew, Arthur Petron, Alex Paino, Matthias Plappert, Glenn Powell, Raphael Ribas, et al. Solving Rubik's Cube With a Robot Hand. *arXiv Preprint*, 2019.

[4] R McNeill Alexander. *Principles of Animal Locomotion*. Princeton University Press, 2003. ISBN 9780691126340.

[5] Lauren W Ancel. A Quantitative Model of the Simpson–Baldwin Effect. *Journal of Theoretical Biology*, 196(2):197–209, 1999.

[6] Lauren W Ancel. Undermining the Baldwin Expediting Effect: Does Phenotypic Plasticity Accelerate Evolution? *Theoretical Population Biology*, 58(4): 307–319, 2000.

[7] W Ross Ashby. *Design for a Brain: The Origin of Adaptive Behaviour*. Chapman & Hall, 1952.

[8] Anish Athalye, Logan Engstrom, Andrew Ilyas, and Kevin Kwok. Synthesizing Robust Adversarial Examples. In *International Conference on Machine Learning*, pages 284–293, 2018.

[9] Joshua E Auerbach and Josh C Bongard. Dynamic Resolution in the Co-Evolution of Morphology and Control. In *Artificial Life XII: Proceedings of the Twelfth International Conference on the Synthesis and Simulation of Living Systems*, pages 451–458. MIT Press, 2010.

[10] Joshua E Auerbach and Josh C Bongard. Evolving CPPNS to Grow Three-Dimensional Physical Structures. In *Proceedings of the 12th Annual Conference on Genetic and Evolutionary Computation*, pages 627–634. ACM, 2010.

[11] Joshua E Auerbach and Josh C Bongard. Environmental Influence on the Evolution of Morphological Complexity in Machines. *PLoS Computational Biology*, 10(1):e1003399, 2014.

[12] J Mark Baldwin. A New Factor in Evolution. *The American Naturalist*, 30 (354):441–451, 1896.

[13] František Baluška and Michael Levin. On Having No Head: Cognition Throughout Biological Systems. *Frontiers in Psychology*, 7:902, 2016.

[14] David Basanta, Mark Miodownik, and Buzz Baum. The Evolution of Robust Development and Homeostasis in Artificial Organisms. *PLoS Computational Biology*, 4(3):e1000030, 2008.

[15] Patrick Bateson. How Do Sensitive Periods Arise and What Are They For? *Animal Behaviour*, 27:470–486, 1979.

[16] CW Beck and JM Slack. A Developmental Pathway Controlling Outgrowth of the Xenopus Tail Bud. *Development*, 126(8):1611–1620, 1999.

[17] Randall D Beer. Toward the Evolution of Dynamical Neural Networks for Minimally Cognitive Behavior. In *Proceedings of the Fourth International Conference on the Simulation of Adaptive Behavior*, 1996.

[18] Stafford Beer. A Filigree Friendship. *Kybernetes*, 2001.

[19] Richard K Belew. Evolution, Learning and Culture: Computational Metaphors for Adaptive Algorithms. *Technical Report CS89-156, Computer Science, Univ. Calif. San. Diego*, 1989.

[20] Maxwell R Bennett, Peter Michael Stephan Hacker, and MR Bennett. *Philosophical Foundations of Neuroscience*, volume 79. Blackwell Oxford, 2003.

[21] Homanga Bharadhwaj, Zihan Wang, Yoshua Bengio, and Liam Paull. A Data-Efficient Framework for Training and Sim-To-Real Transfer of Navigation Policies. *arXiv Preprint*, 2018.

[22] Jon Bird and Paul Layzell. The Evolved Radio and Its Implications for Modelling the Evolution of Novel Sensors. In *Proceedings of the 2002 Congress on Evolutionary Computation (CEC)*, pages 1836–1841. IEEE, 2002.

[23] Douglas J Blackiston and Michael Levin. Ectopic Eyes Outside the Head in Xenopus Tadpoles Provide Sensory Data for Light-Mediated Learning. *Journal of Experimental Biology*, 216(6):1031–1040, 2013.

[24] Douglas J Blackiston, Elena Silva Casey, and Martha R Weiss. Retention of Memory Through Metamorphosis: Can a Moth Remember What It Learned as a Caterpillar? *PLoS One*, 3(3):e1736, 2008.

[25] Douglas J Blackiston, Tal Shomrat, and Michael Levin. The Stability of Memories During Brain Remodeling: A Perspective. *Communicative & Integrative Biology*, 8(5):e1073424, 2015.

[26] Josh Bongard. The Utility of Evolving Simulated Robot Morphology Increases With Task Complexity for Object Manipulation. *Artificial Life*, 16(3):201–223, 2010.

[27] Josh Bongard, Victor Zykov, and Hod Lipson. Resilient Machines Through Continuous Self-Modeling. *Science*, 314(5802):1118–1121, 2006.

[28] Josh C Bongard. Morphological Change in Machines Accelerates the Evolution of Robust Behavior. *Proceedings of the National Academy of Sciences*, 108(4): 1234–1239, 2011.

[29] Josh C Bongard and Rolf Pfeifer. Repeated Structure and Dissociation of Genotypic and Phenotypic Complexity in Artificial Ontogeny. *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO)*, pages 829–836, 2001.

[30] Josh C Bongard and Rolf Pfeifer. Evolving Complete Agents Using Artificial Ontogeny. *Morpho-Functional Machines: The New Species (Designing Embodied Intelligence)*, pages 237–258, 2003.

[31] Joran W Booth, Jennifer C Case, Edward L White, Dylan S Shah, and Rebecca Kramer-Bottiglio. An Addressable Pneumatic Regulator for Distributed Control of Soft Robots. In *2018 IEEE International Conference on Soft Robotics (RoboSoft)*, pages 25–30, 2018.

[32] Jeremy P Brockes. Amphibian Limb Regeneration: Rebuilding a Complex Structure. *Science*, 276(5309):81–87, 1997.

[33] Luzius Brodbeck, Simon Hauser, and Fumiya Iida. Morphological Evolution of Physical Robots Through Model-Free Phenotype Development. *PloS One*, 10 (6):e0128444, 2015.

[34] Eric Brown, Nicholas Rodenberg, John Amend, Annan Mozeika, Erik Steltz, Mitchell R Zakin, Hod Lipson, and Heinrich M Jaeger. Universal Robotic Gripper Based on the Jamming of Granular Material. *Proceedings of the National Academy of Sciences*, 107(44):18809–18814, 2010.

[35] Ken Caluwaerts, Jérémie Despraz, Atıl Işçen, Andrew P Sabelhaus, Jonathan Bruce, Benjamin Schrauwen, and Vytas SunSpiral. Design and Control of Compliant Tensegrity Robots Through Simulation and Hardware Validation. *Journal of the Royal Society Interface*, 11(98):20140520, 2014.

[36] Peter Cariani. To Evolve an Ear. Epistemological Implications of Gordon Pask's Electrochemical Devices. *Systems Research*, 10(3):19–33, 1993.

[37] Jennifer Carlson and Robin R Murphy. How UGVs Physically Fail in the Field. *IEEE Transactions on Robotics*, 21(3):423–437, 2005.

[38] Daniel Cellucci, Robert MacCurdy, Hod Lipson, and Sebastian Risi. 1D Printing of Recyclable Robots. *IEEE Robotics and Automation Letters*, 2(4):1964–1971, 2017.

[39] Konstantinos Chatzilygeroudis, Vassilis Vassiliades, and Jean-Baptiste Mouret. Reset-Free Trial-And-Error Learning for Robot Damage Recovery. *Robotics and Autonomous Systems*, 100:236–250, 2018.

[40] Nicolas Chaumont, Richard Egli, and Christoph Adami. Evolving Virtual Creatures and Catapults. *Artificial Life*, 13(2):139–157, 2007.

[41] Yevgen Chebotar, Ankur Handa, Viktor Makoviychuk, Miles Macklin, Jan Issac, Nathan Ratliff, and Dieter Fox. Closing the Sim-To-Real Loop: Adapting Simulation Randomization With Real World Experience. In *The International Conference on Robotics and Automation (ICRA)*, pages 8973–8979. IEEE, 2019.

[42] Nick Cheney, Robert MacCurdy, Jeff Clune, and Hod Lipson. Unshackling Evolution: Evolving Soft Robots With Multiple Materials and a Powerful Generative Encoding. In *Proceedings of the 15th Annual Conference on Genetic and Evolutionary Computation*, pages 167–174. ACM, 2013.

[43] Nick Cheney, Jeff Clune, and Hod Lipson. Evolved Electrophysiological Soft Robots. In *Proceedings of the Conference on Artificial Life (ALife)*, volume 14, pages 222–229, 2014.

[44] Nick Cheney, Josh C Bongard, and Hod Lipson. Evolving Soft Robots in Tight Spaces. In *Proceedings of the 2015 Annual Conference on Genetic and Evolutionary Computation*, pages 935–942. ACM, 2015.

[45] Nick Cheney, Josh Bongard, Vytas SunSpiral, and Hod Lipson. On the Difficulty of Co-Optimizing Morphology and Control in Evolved Virtual Creatures. In *Proceedings of the Artificial Life Conference*, pages 226–234, 2016.

[46] Nick Cheney, Josh Bongard, Vytas SunSpiral, and Hod Lipson. Scalable Co-Optimization of Morphology and Control in Embodied Machines. *Journal of the Royal Society Interface*, 15(143):20170937, 2018.

[47] Andy Clark. Whatever Next? Predictive Brains, Situated Agents, and the Future of Cognitive Science. *Behavioral and Brain Sciences*, 36(3):181–204, 2013.

[48] Dave Cliff, Phil Husbands, and Inman Harvey. Explorations in Evolutionary Robotics. *Adaptive Behavior*, 2(1):73–110, 1993.

[49] Jeff Clune, Benjamin E Beckmann, Charles Ofria, and Robert T Pennock. Evolving Coordinated Quadruped Gaits With the Hyperneat Generative Encoding. In *The IEEE Congress on Evolutionary Computation*, pages 2764–2771, 2009.

[50] Steve Collins, Andy Ruina, Russ Tedrake, and Martijn Wisse. Efficient Bipedal Robots Based on Passive-Dynamic Walkers. *Science*, 307(5712):1082–1085, 2005.

[51] Roger C Conant and W Ross Ashby. Every Good Regulator of a System Must Be a Model of That System. *International Journal of Systems Science*, 1(2): 89–97, 1970.

[52] Jonathan Cooke. Scale of Body Pattern Adjusts to Available Cell Number in Amphibian Embryos. *Nature*, 290(5809):775–778, 1981.

[53] William C Corning. Regeneration and Retention of Acquired Information. In *Chemistry of Learning*, pages 281–294. Springer, 1967.

[54] Francesco Corucci, Nick Cheney, Sam Kriegman, Josh Bongard, and Cecilia Laschi. Evolutionary Developmental Soft Robotics as a Framework to Study Intelligence and Adaptive Behavior in Animals and Plants. *Frontiers in Robotics and AI*, 4:34, 2017.

[55] Antoine Cully, Jeff Clune, Danesh Tarapore, and Jean-Baptiste Mouret. Robots That Can Adapt Like Animals. *Nature*, 521:503–507, 2015.

[56] Caroline Cvetkovic, Ritu Raman, Vincent Chan, Brian J Williams, Madeline Tolish, Piyush Bajaj, Mahmut Selman Sakar, H Harry Asada, M Taher A Saif, and Rashid Bashir. Three-Dimensionally Printed Biological Machines Powered by Skeletal Muscle. *Proceedings of the National Academy of Sciences*, 111(28): 10125–10130, 2014.

[57] Richard Dawkins. *The Extended Phenotype: The Long Reach of the Gene*. Oxford University Press, 1982.

[58] Gisèle A Deblandre, Daniel A Wettstein, Naoko Koyano-Nakagawa, and Chris Kintner. A Two-Step Mechanism Generates the Spacing Pattern of the Ciliated Cells in the Skin of Xenopus Embryos. *Development*, 126(21):4715–4728, 1999.

[59] Frank Dellaert and Randall D Beer. Toward an Evolvable Model of Development for Autonomous Agent Synthesis. In *Artificial Life IV, Proceedings of the Fourth International Workshop on the Synthesis and Simulation of Living Systems*, pages 246–257. MIT Press, 1994.

[60] Frank Dellaert and Randall D Beer. A Developmental Model for the Evolution of Complete Autonomous Agents. In *Proceedings of the Fourth International Conference on Simulation of Adaptive Behavior*, 1996.

[61] Daniel C Dennett. Why the Law of Effect Will Not Go Away. *Journal for the Theory of Social Behaviour*, 1975.

[62] Daniel C Dennett. The Baldwin Effect: A Crane, Not a Skyhook. *Evolution and Learning: The Baldwin Effect Reconsidered*, pages 60–79, 2003.

[63] R Vasundhara Devi, S Siva Sathya, and Mohane Selvaraj Coumar. Evolutionary Algorithms for De Novo Drug Design–A Survey. *Applied Soft Computing*, 27: 543–552, 2015.

[64] René Doursat. Organically Grown Architectures: Creating Decentralized, Autonomous Systems by Embryomorphic Engineering. In *Organic Computing*, pages 167–199. Springer, 2009.

[65] René Doursat and Carlos Sánchez. Growing Fine-Grained Multicellular Robots. *Soft Robotics*, 1(2):110–121, 2014.

[66] Keith L Downing. Development and the Baldwin Effect. *Artificial Life*, 10(1): 39–63, 2004.

[67] Hubert L Dreyfus. Why Computers Must Have Bodies in Order to Be Intelligent. *The Review of Metaphysics*, pages 13–32, 1967.

[68] Peter Eggenberger. Evolving Morphologies of Simulated 3D Organisms Based on Differential Gene Expression. In *Proceedings of the European Conference on Artificial Life*, pages 205–213, 1997.

[69] Thomas Elsken, Jan Hendrik Metzen, and Frank Hutter. Neural Architecture Search: A Survey. *arXiv Preprint arXiv:1808.05377*, 2018.

[70] Andrés Faíña, Francisco Bellas, Fernando López-Peña, and Richard J Duro. EDHMoR: Evolutionary Designer of Heterogeneous Modular Robots. *Engineering Applications of Artificial Intelligence*, 26(10):2408–2423, 2013.

[71] Tim W. Fawcett and Willem E. Frankenhuis. Adaptive explanations for sensitive windows in development. *Frontiers in Zoology*, 12(1):S3, Aug 2015. ISSN 1742-9994.

[72] Adam W Feinberg, Alex Feigel, Sergey S Shevkoplyas, Sean Sheehy, George M Whitesides, and Kevin Kit Parker. Muscular Thin Films for Building Actuators and Powering Devices. *Science*, 317(5843):1366–1370, 2007.

[73] Samuel Felton, Michael Tolley, Erik Demaine, Daniela Rus, and Robert Wood. A Method for Building Self-Folding Machines. *Science*, 345(6197):644–646, 2014.

[74] Chrisantha Thomas Fernando, Jakub Sygnowski, Simon Osindero, Jane Wang, Tom Schaul, Denis Teplyashin, Pablo Sprechmann, Alexander Pritzel, and Andrei A Rusu. Meta Learning by the Baldwin Effect. *arXiv Preprint arXiv:1806.07917*, 2018.

[75] Chris Fields and Michael Levin. Scale-Free Biology: Integrating Evolutionary and Developmental Thinking. *BioEssays*, 42(8):1900228, 2020.

[76] Ronald Aylmer Fisher. *The Genetical Theory of Natural Selection*. Oxford University Press, 1930.

[77] D Floreano and F Mondada. Evolution of Plastic Neurocontrollers for Situated Agents. In *From Animals to Animats 4, Proceedings of the 4th International Conference on Simulation of Adaptive Behavior (SAB)*, pages 402–410. MIT Press, 1996.

[78] Dario Floreano and Francesco Mondada. Evolution of Homing Navigation in a Real Mobile Robot. *IEEE Transactions on Systems, Man, and Cybernetics*, 26 (3):396–407, 1996.

[79] David B Fogel. *Evolutionary Computation: The Fossil Record*. IEEE Press, 1998.

[80] Robert French and Adam Messinger. Genes, Phenes and the Baldwin Effect: Learning and Evolution in a Simulated Population. In *Artificial Life IV*, pages 277–282. Cambridge, MA: MIT Press, 1994.

[81] Robert M French. Dynamically Constraining Connectionist Networks to Produce Distributed, Orthogonal Representations to Reduce Catastrophic Interference. In *Proceedings of the 16th Annual Cog. Sci. Society Conference*, 1994.

[82] Robert M French. Catastrophic Forgetting in Connectionist Networks. *Trends in Cognitive Sciences*, 3(4):128–135, 1999.

[83] A Sydney Gladman, Elisabetta A Matsumoto, Ralph G Nuzzo, L Mahadevan, and Jennifer A Lewis. Biomimetic 4d Printing. *Nature Materials*, 15(4):413–418, 2016.

[84] Florian Golemo, Adrien Ali Taiga, Aaron Courville, and Pierre-Yves Oudeyer. Sim-To-Real Transfer With Neural-Augmented Robot Simulation. In *Proceedings of the 2nd Conference on Robot Learning*, volume 87 of *Proceedings of Machine Learning Research*, pages 817–828. PMLR, 2018.

[85] Ian Goodfellow, Mehdi Mirza, Da Xiao, Aaron Courville, and Yoshua Bengio. An Empirical Investigation of Catastrophic Forgetting in Gradient-Based Neural Networks. *arXiv Preprint arXiv:1312.6211*, 2013.

[86] Martyn Goulding. Circuits Controlling Vertebrate Locomotion: Moving in a New Direction. *Nature Reviews Neuroscience*, 10(7):507, 2009.

[87] Gail L Grabowsky. Symmetry, Locomotion, and the Evolution of an Anterior End: A Lesson From Sea Urchins. *Evolution*, 48(4):1130–1146, 1994.

[88] David Ha. Evolving Stable Strategies, 2017. blog.otoro.net/2017/11/12/evolving-stable-strategies.

[89] I. Harvey, P. Husbands, D. Cliff, A. Thompson, and N. Jakobi. Evolutionary robotics: the sussex approach. *Robotics and Autonomous Systems*, 20(2):205 – 224, 1997. ISSN 0921-8890.

[90] Inman Harvey. Robotics: Philosophy of Mind Using a Screwdriver. In *Evolutionary Robotics: From Intelligent Robots to Artificial Life, Vol. III*, pages 207–230, 2000.

[91] Inman Harvey, Phil Husbands, Dave Cliff, Adrian Thompson, and Nick Jakobi. Evolutionary Robotics: The Sussex Approach. *Robotics and Autonomous Systems*, 20(2-4):205–224, 1997.

[92] Elliot Hawkes, B An, Nadia M Benbernou, H Tanaka, Sangbae Kim, ED Demaine, D Rus, and Robert J Wood. Programmable Matter by Folding. *Proceedings of the National Academy of Sciences*, 107(28):12441–12445, 2010.

[93] Hugh Herr and Robert G Dennis. A Swimming Robot Actuated by Living Muscle Tissue. *Journal of Neuroengineering and Rehabilitation*, 1(1):1–9, 2004.

[94] Jonathan Hiller and Hod Lipson. Automatic Design and Manufacture of Soft Robots. *IEEE Transactions on Robotics*, 28(2):457–466, 2012.

[95] Jonathan Hiller and Hod Lipson. Dynamic Simulation of Soft Multimaterial 3D-Printed Objects. *Soft Robotics*, 1(1):88–101, 2014.

[96] Geoffrey E Hinton and Steven J Nowlan. How Learning Can Guide Evolution. *Complex Systems*, 1(3):495–502, 1987.

[97] Daniel Holden, Bang Chi Duong, Sayantan Datta, and Derek Nowrouzezahrai. Subspace Neural Physics: Fast Data-Driven Interactive Simulation. In *Proceedings of the 18th Annual ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, page 6, 2019.

[98] Gregory S Hornby and Jordan B Pollack. Creating High-Level Components With a Generative Representation for Body-Brain Evolution. *Artificial Life*, 8 (3):223–246, 2002.

[99] Gregory S Hornby, Hod Lipson, and Jordan B Pollack. Generative Representations for the Automated Design of Modular Physical Robots. *IEEE Transactions on Robotics and Automation*, 19(4):703–719, 2003.

[100] Peter Eggenberger Hotz. Asymmetric Cell Division and Its Integration With Other Developmental Processes for Artificial Evolutionary Systems. In *Artificial Life IX: Proceedings of the Ninth International Conference on the Simulation and Synthesis of Artificial Life*, volume 9, page 387. MIT Press, 2004.

[101] Mark D Huntington, Lincoln J Lauhon, and Teri W Odom. Subwavelength Lattice Optics by Evolutionary Design. *Nano Letters*, 14(12):7195–7200, 2014.

[102] Phil Husbands, Tom Smith, Nick Jakobi, and Michael O'Shea. Better Living Through Chemistry: Evolving Gasnets for Robot Control. *Connection Science*, 10(3-4):185–210, 1998.

[103] Clyde A Hutchison, Ray-Yuan Chuang, Vladimir N Noskov, Nacyra Assad-Garcia, Thomas J Deerinck, Mark H Ellisman, John Gill, Krishna Kannan,

Bogumil J Karas, Li Ma, et al. Design and Synthesis of a Minimal Bacterial Genome. *Science*, 351(6280), 2016.

[104] Jemin Hwangbo, Joonho Lee, Alexey Dosovitskiy, Dario Bellicoso, Vassilios Tsounis, Vladlen Koltun, and Marco Hutter. Learning Agile and Dynamic Motor Skills for Legged Robots. *Science Robotics*, 4(26), 2019.

[105] Cynthia M Illingworth. Trapped Fingers and Amputated Finger Tips in Children. *Journal of Pediatric Surgery*, 9(6):853–858, 1974.

[106] Max Jaderberg, Wojciech M Czarnecki, Iain Dunning, Luke Marris, Guy Lever, Antonio Garcia Castaneda, Charles Beattie, Neil C Rabinowitz, Ari S Morcos, Avraham Ruderman, et al. Human-Level Performance in 3D Multiplayer Games With Population-Based Reinforcement Learning. *Science*, 364(6443):859–865, 2019.

[107] Nick Jakobi. Evolutionary Robotics and the Radical Envelope-Of-Noise Hypothesis. *Adaptive Behavior*, 6(2):325–368, 1997.

[108] Nick Jakobi, Phil Husbands, and Inman Harvey. Noise and the Reality Gap: The Use of Simulation in Evolutionary Robotics. In *European Conference on Artificial Life*, pages 704–720. Springer, 1995.

[109] Michał Joachimczak, Reiji Suzuki, and Takaya Arita. Artificial Metamorphosis: Evolutionary Design of Transforming, Soft-Bodied Robots. *Artificial Life*, 22 (3):271–298, 2016.

[110] Roger D Kamm, Rashid Bashir, Natasha Arora, Roy D Dar, Martha U Gillette, Linda G Griffith, Melissa L Kemp, Kathy Kinlaw, Michael Levin, Adam C Martin, et al. Perspective: The Promise of Multi-Cellular Engineered Living Systems. *APL Bioengineering*, 2(4):040901, 2018.

[111] Takeshi Kano, Eiki Sato, Tatsuya Ono, Hitoshi Aonuma, Yoshiya Matsuzaka, and Akio Ishiguro. A Brittle Star-Like Robot Capable of Immediately Adapting to Unexpected Physical Damage. *Royal Society Open Science*, 4(12):171200, 2017.

[112] Diederik P Kingma and Jimmy Ba. Adam: A Method for Stochastic Optimization. *arXiv Preprint arXiv:1412.6980*, 2014.

[113] Alexander S Klyubin, Daniel Polani, and Chrystopher L Nehaniv. Empowerment: A Universal Agent-Centric Measure of Control. In *2005 IEEE Congress on Evolutionary Computation*, volume 1, pages 128–135, 2005.

[114] Maciej Komosinski. The Framsticks System: Versatile Simulator of 3D Agents and Their Evolution. *Kybernetes*, 32(1/2):156–173, 2003.

[115] Maciej Komosiński and Adam Rotaru-Varga. Comparison of Different Genotype Encodings for Simulated Three-Dimensional Agents. *Artificial Life*, 7(4):395–418, 2001.

[116] Sylvain Koos, Jean-Baptiste Mouret, and Stéphane Doncieux. The Transferability Approach: Crossing the Reality Gap in Evolutionary Robotics. *IEEE Transactions on Evolutionary Computation*, 17(1):122–145, 2012.

[117] Sylvain Koos, Jean-Baptiste Mouret, and Stéphane Doncieux. The Transferability Approach: Crossing the Reality Gap in Evolutionary Robotics. *IEEE Transactions on Evolutionary Computation*, 17(1):122–145, 2013.

[118] Arda Kotikian, Connor McMahan, Emily C Davidson, Jalilah M Muhammad, Robert D Weeks, Chiara Daraio, and Jennifer A Lewis. Untethered Soft Robotic Matter With Passive Control of Shape Morphing and Propulsion. *Science Robotics*, 4(33):eaax7044, 2019.

[119] Kostas Kouvaris, Jeff Clune, Loizos Kounios, Markus Brede, and Richard Watson. How Evolution Learns to Generalise: Using the Principles of Learning Theory to Understand the Evolution of Developmental Organisation. *PLoS Computational Biology*, pages 1–41, 2017.

[120] Sam Kriegman. Why Virtual Creatures Matter. *Nature Machine Intelligence*, 1(10):492–492, 2019.

[121] Sam Kriegman, Nick Cheney, Francesco Corucci, and Josh C Bongard. A Minimal Developmental Model Can Increase Evolvability in Soft Robots. In *Proceedings of the Genetic and Evolutionary Computation Conference*, pages 131–138. ACM, 2017.

[122] Sam Kriegman, Nick Cheney, and Josh Bongard. How Morphological Development Can Guide Evolution. *Scientific Reports*, 8(1):13934, 2018.

[123] Sam Kriegman, Nick Cheney, Francesco Corucci, and Josh C. Bongard. Interoceptive Robustness Through Environment-Mediated Morphological Development. In *Proceedings of the Genetic and Evolutionary Computation Conference*, pages 109–116. ACM, 2018.

[124] Sam Kriegman, Stephanie Walker, Dylan Shah, Michael Levin, Rebecca Kramer-Bottiglio, and Josh Bongard. Automated Shapeshifting for Function

Recovery in Damaged Robots. In *Proceedings of Robotics: Science and Systems (RSS)*, 2019.

[125] Sam Kriegman, Douglas Blackiston, Michael Levin, and Josh Bongard. A Scalable Pipeline for Designing Reconfigurable Organisms. *Proceedings of the National Academy of Sciences*, 117(4):1853–1859, 2020.

[126] Sam Kriegman, Amir Mohammadi Nasab, Dylan Shah, Hannah Steele, Gabrielle Branin, Michael Levin, Josh Bongard, and Rebecca Kramer-Bottiglio. Scalable Sim-To-Real Transfer of Soft Robot Designs. In *Proceedings of the 3rd IEEE International Conference on Soft Robotics (RoboSoft)*, 2020.

[127] Robert Kwiatkowski and Hod Lipson. Task-Agnostic Self-Modeling Machines. *Science Robotics*, 4(26), 2019.

[128] Russell Lande. Adaptation to an Extraordinary Environment by Evolution of Phenotypic Plasticity and Genetic Assimilation. *Journal of Evolutionary Biology*, 22(7):1435–1446, 2009.

[129] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep Learning. *Nature*, 521(7553):436–444, 2015.

[130] Joel Lehman and Kenneth O Stanley. Evolving a Diversity of Virtual Creatures Through Novelty Search and Local Competition. In *Proceedings of the 13th Annual Conference on Genetic and Evolutionary Computation*, pages 211–218, 2011.

[131] Joel Lehman, Jay Chen, Jeff Clune, and Kenneth O Stanley. Safe Mutations for Deep and Recurrent Neural Networks Through Output Gradients. In *Proceedings of the Genetic and Evolutionary Computation Conference*, pages 117–124, 2018.

[132] Dan Lessin and Sebastian Risi. Soft-Body Muscles for Evolved Virtual Creatures: The Next Step on a Bio-Mimetic Path to Meaningful Morphological Complexity. In *Artificial Life Conference Proceedings 13*, pages 604–611, 2015.

[133] Dan Lessin, Don Fussell, and Risto Miikkulainen. Open-Ended Behavioral Complexity for Evolved Virtual Creatures. In *Proceedings of the 15th Annual Conference on Genetic and Evolutionary Computation*, pages 335–342, 2013.

[134] Michael Levin. Reprogramming cells and tissue patterning via bioelectrical pathways: molecular mechanisms and biomedical opportunities. *Wiley Interdisciplinary Reviews: Systems Biology and Medicine*, 5(6):657–676, 2013.

[135] Jinxing Li, Berta Esteban-Fernández de Ávila, Wei Gao, Liangfang Zhang, and Joseph Wang. Micro/Nanorobots for Biomedicine: Delivery, Surgery, Sensing, and Detoxification. *Science Robotics*, 2(4), 2017.

[136] Shuguang Li, Richa Batra, David Brown, Hyun-Dong Chang, Nikhil Ranganathan, Chuck Hoberman, Daniela Rus, and Hod Lipson. Particle Robotics Based on Statistical Mechanics of Loosely Coupled Components. *Nature*, 567 (7748):361–365, 2019.

[137] Lukas Lichtensteiger and Peter Eggenberger. Evolving the Morphology of a Compound Eye on a Robot. In *The Third European Workshop on Advanced Mobile Robots (Eurobot)*, pages 127–134. IEEE, 1999.

[138] Hod Lipson. Principles of Modularity, Regularity, and Hierarchy for Scalable Systems. *The Journal of Biological Physics and Chemistry*, 7(4):125–128, 2007.

[139] Hod Lipson. Challenges and Opportunities for Design, Simulation, and Fabrication of Soft Robots. *Soft Robotics*, 1(1):21–27, 2014.

[140] Hod Lipson and Jordan B Pollack. Automatic Design and Manufacture of Robotic Lifeforms. *Nature*, 406(6799):974, 2000.

[141] Sida Liu, David Matthews, Sam Kriegman, and Josh Bongard. Voxcraft-Sim, a GPU-accelerated Voxel-Based Physics Engine. github.com/voxcraft/voxcraft-sim, 2020. 10.5281/zenodo.3835152.

[142] Daniel Lobo, Wendy S Beane, and Michael Levin. Modeling Planarian Regeneration: A Primer for Reverse-Engineering the Worm. *PLoS Computational Biology*, 8(4):e1002481, 2012.

[143] Michael Lynch. Evolution of the Mutation Rate. *Trends in Genetics*, 26(8): 345–352, 2010.

[144] Miles Macklin, Kenny Erleben, Matthias Müller, Nuttapong Chentanez, Stefan Jeschke, and Viktor Makoviychuk. Non-Smooth Newton Methods for Deformable Multi-Body Dynamics. *ACM Transactions on Graphics (TOG)*, 38 (5):140, 2019.

[145] John D Madden, Serge R Lafontaine, and Ian W Hunter. Fabrication by Electrodeposition: Building 3D Structures and Polymer Actuators. In *Proceedings of the Sixth International Symposium on Micro Machine and Human Science*, pages 77–81. IEEE, 1995.

[146] Siavash Haroun Mahdavi and Peter J Bentley. An Evolutionary Approach to Damage Recovery of Robot Motion With Muscles. In *European Conference on Artificial Life*, pages 248–255. Springer, 2003.

[147] Carmel Majidi, Robert F Shepherd, Rebecca K Kramer, George M Whitesides, and Robert J Wood. Influence of Surface Traction on Soft Robot Undulation. *The International Journal of Robotics Research*, 32(13):1577–1584, 2013.

[148] Carlo C Maley. Four Steps Toward Open-Ended Evolution. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO)*, volume 2, pages 1336–1343, 1999.

[149] Jan Matas, Stephen James, and Andrew J Davison. Sim-To-Real Reinforcement Learning for Deformable Object Manipulation. In *Proceedings of the 2nd Conference on Robot Learning*, volume 87 of *Proceedings of Machine Learning Research*, pages 734–743. PMLR, 2018.

[150] Giles Mayley. Landscapes, Learning Costs, and Genetic Assimilation. *Evolutionary Computation*, 4(3):213–234, 1996.

[151] Catherine McCusker and David M Gardiner. The Axolotl Model for Regeneration and Aging Research: A Mini-Review. *Gerontology*, 57(6):565–571, 2011.

[152] Orazio Miglino, Kourosh Nafasi, and Charles E. Taylor. Selection for wandering behavior in a small robot. *Artificial Life*, 2(1):101–116, 1994.

[153] Risto Miikkulainen, Jason Liang, Elliot Meyerson, Aditya Rawal, Dan Fink, Olivier Francon, Bala Raju, Arshak Navruzyan, Nigel Duffy, and Babak Hodjat. Evolving Deep Neural Networks. *arXiv Preprint arXiv:1703.00548*, 2017.

[154] Julian Francis Miller. Evolving a Self-Repairing, Self-Regulating, French Flag Organism. In *Genetic and Evolutionary Computation Conference*, pages 129–139. Springer, 2004.

[155] Aslan Miriyev, Kenneth Stack, and Hod Lipson. Soft Material for Soft Actuators. *Nature Communications*, 8(1):596, 2017.

[156] Shuhei Miyashita, Steven Guitron, Shuguang Li, and Daniela Rus. Robotic Metamorphosis by Origami Exoskeletons. *Science Robotics*, 2(10):eaao4369, 2017.

[157] Armin P Moczek, Sonia Sultan, Susan Foster, Cris Ledón-Rettig, Ian Dworkin, H Fred Nijhout, Ehab Abouheif, and David W Pfennig. The Role of Developmental Plasticity in Evolutionary Innovation. *Proceedings of the Royal Society of London B: Biological Sciences*, 278(1719):2705–2713, 2011.

[158] Rico Moeckel, Yura N Perov, Anh The Nguyen, Massimo Vespignani, Stéphane Bonardi, Soha Pouya, Alexander Sproewitz, Jesse van den Kieboom, Frédéric Wilhelm, and Auke Jan Ijspeert. Gait Optimization for Roombots Modular Robots–Matching Simulation and Reality. In *The IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3265–3272, 2013.

[159] JR Montgomery and SJ Coward. On the Minimal Size of a Planarian Capable of Regeneration. *Transactions of the American Microscopical Society*, 93(3): 386, 1974.

[160] Alexander Mordvintsev, Ettore Randazzo, Eyvind Niklasson, and Michael Levin. Growing Neural Cellular Automata. *Distill*, 5(2):e23, 2020.

[161] C Lloyd Morgan. On Modification and Variation. *Science*, 4(99):733–740, 1896.

[162] Stephen A. Morin, Sen Wai Kwok, Joshua Lessing, Jason Ting, Robert F. Shepherd, Adam A. Stokes, and George M. Whitesides. Elastomeric Tiles for the Fabrication of Inflatable Structures. *Advanced Functional Materials*, 24(35): 5541–5549, 2014.

[163] Caitlin T Mueller and John A Ochsendorf. Combining Structural Performance and Designer Preferences in Evolutionary Design Space Exploration. *Automation in Construction*, 52:70–82, 2015.

[164] David J Munk, Gareth A Vio, and Grant P Steven. Topology and Shape Optimization Methods Using Evolutionary Algorithms: A Review. *Structural and Multidisciplinary Optimization*, 52(3):613–631, 2015.

[165] Courtney J Murren, Josh R Auld, H Callahan, Cameron K Ghalambor, Corey A Handelsman, Mary A Heskel, JG Kingsolver, Heidi J Maclean, Joanna Masel, Heather Maughan, et al. Constraints on the Evolution of Phenotypic Plasticity: Limits and Costs of Phenotype and Plasticity. *Heredity*, 115(4):293–301, 2015.

[166] Ofir Nachum, Michael Ahn, Hugo Ponte, Shixiang Gu, and Vikash Kumar. Multi-Agent Manipulation via Locomotion Using Hierarchical Sim2real. In *Proceedings of the 3rd Conference on Robot Learning*, 2019.

[167] Yashraj S Narang, Alperen Degirmenci, Joost J Vlassak, and Robert D Howe. Transforming the Dynamic Response of Robotic Structures and Systems Through Laminar Jamming. *IEEE Robotics and Automation Letters*, 3 (2):688–695, 2018.

[168] NASA. Nuclear and Space Radiation Effects on Materials. *Space Vehicle Design Criteria*, SP-8053, 1970.

[169] Janna C Nawroth, Hyungsuk Lee, Adam W Feinberg, Crystal M Ripplinger, Megan L McCain, Anna Grosberg, John O Dabiri, and Kevin Kit Parker. A Tissue-Engineered Jellyfish With Biomimetic Propulsion. *Nature Biotechnology*, 30(8):792–797, 2012.

[170] Anh Nguyen, Jason Yosinski, and Jeff Clune. Deep Neural Networks Are Easily Fooled: High Confidence Predictions for Unrecognizable Images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 427–436, 2015.

[171] Pieter D Nieuwkoop. Normal Table of Xenopus Laevis (Daudin). *Normal Table of Xenopus Laevis (Daudin)*, pages 162–203, 1956.

[172] Stefano Nolfi and Dario Floreano. *Evolutionary Robotics: The Biology, Intelligence, and Technology of Self-Organizing Machines*. MIT press, 2000.

[173] Stefano Nolfi, Domenico Parisi, and Jeffrey L Elman. Learning and Evolution in Neural Networks. *Adaptive Behavior*, 3(1):5–28, 1994.

[174] Tønnes F Nygaard, Charles P Martin, Eivind Samuelsen, Jim Torresen, and Kyrre Glette. Real-World Evolution Adapts Robot Morphology and Control to Hardware Limitations. In *Proceedings of the Genetic and Evolutionary Computation Conference*, pages 125–132, 2018.

[175] Tønnes F Nygaard, Charles P Martin, David Howard, Jim Torresen, and Kyrre Glette. Environmental Adaptation of Robot Morphology and Control Through Real-World Evolution. *arXiv Preprint arXiv:2003.13254*, 2020.

[176] Sung-Jin Park, Mattia Gazzola, Kyung Soo Park, Shirley Park, Valentina Di Santo, Erin L Blevins, Johan U Lind, Patrick H Campbell, Stephanie Dauth, Andrew K Capulli, et al. Phototactic Guidance of a Tissue-Engineered Soft-Robotic Ray. *Science*, 353(6295):158–162, 2016.

[177] Gordon Pask. Physical Analogues to the Growth of a Concept. In *Mechanization of Thought Processes, Symposium*, volume 10, pages 765–794, 1958.

[178] Gordon Pask. The Natural History of Networks. *Self-Organizing Systems*, pages 232–263, 1960.

[179] Gordon Pask. *An Approach to Cybernetics*. Hutchinson, 1961.

[180] Deepak Pathak, Chris Lu, Trevor Darrell, Phillip Isola, and Alexei A Efros. Learning to Control Self-Assembling Morphologies: A Study of Generalization via Modularity. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

[181] Debabrata Patra, Samudra Sengupta, Wentao Duan, Hua Zhang, Ryan Pavlick, and Ayusman Sen. Intelligent, Self-Powered, Drug Delivery Systems. *Nanoscale*, 5(4):1273–1283, 2013.

[182] Chandana Paul, Francisco J Valero-Cuevas, and Hod Lipson. Design and Control of Tensegrity Robots for Locomotion. *IEEE Transactions on Robotics*, 22 (5):944–957, 2006.

[183] Xue Bin Peng, Marcin Andrychowicz, Wojciech Zaremba, and Pieter Abbeel. Sim-To-Real Transfer of Robotic Control With Dynamics Randomization. In *The IEEE International Conference on Robotics and Automation (ICRA)*, 2018.

[184] Peter Pesic. Shapes of Proteus in Renaissance art. *Huntington Library Quarterly*, 73(1):57–82, 2010.

[185] Giovanni Pezzulo and Michael Levin. Re-Membering the Body: Applications of Computational Neuroscience to the Top-Down Control of Regeneration of Limbs and Other Complex Organs. *Integrative Biology*, 7(12):1487–1517, 2015.

[186] Giovanni Pezzulo and Michael Levin. Top-Down Models in Biology: Explanation and Control of Complex Living Systems Above the Molecular Level. *Journal of the Royal Society Interface*, 13(124):20160555, 2016.

[187] Rolf Pfeifer and Josh Bongard. *How the Body Shapes the Way We Think: A New View of Intelligence*. MIT press, 2006.

[188] Rolf Pfeifer, Max Lungarella, and Fumiya Iida. The Challenges Ahead for Bio-Inspired 'Soft' Robotics. *Communications of the ACM*, 55(11):76–87, 2012.

[189] Lerrel Pinto, Marcin Andrychowicz, Peter Welinder, Wojciech Zaremba, and Pieter Abbeel. Asymmetric Actor Critic for Image-Based Robot Learning. In *Proceedings of Robotics: Science and Systems*, 2018.

[190] Jérôme Piquereau and Renée Ventura-Clapier. Maturation of Cardiac Energy Metabolism During Perinatal Development. *Frontiers in Physiology*, 9:959, 2018.

[191] Samyam Rajbhandari, Jeff Rasley, Olatunji Ruwase, and Yuxiong He. Zero: Memory Optimization Towards Training a Trillion Parameter Models. *arXiv Preprint arXiv:1910.02054*, 2019.

[192] Ritu Raman, Caroline Cvetkovic, Sebastien GM Uzel, Randall J Platt, Parijat Sengupta, Roger D Kamm, and Rashid Bashir. Optogenetic Skeletal Muscle-Powered Adaptive Biological Machines. *Proceedings of the National Academy of Sciences*, 113(13):3497–3502, 2016.

[193] Thomas S Ray. Aesthetically Evolved Virtual Pets. *Leonardo*, 34(4):313–316, 2001.

[194] Guanjiao Ren, Weihai Chen, Sakyasingha Dasgupta, Christoph Kolodziejski, Florentin Wörgötter, and Poramate Manoonpong. Multiple Chaotic Central Pattern Generators With Learning for Legged Locomotion and Malfunction Compensation. *Information Sciences*, 294:666–682, 2015.

[195] Leonardo Ricotti, Barry Trimmer, Adam W Feinberg, Ritu Raman, Kevin K Parker, Rashid Bashir, Metin Sitti, Sylvain Martel, Paolo Dario, and Arianna Menciassi. Biohybrid Actuators for Robotics: A Review of Devices Actuated by Living Cells. *Science Robotics*, 2(12), 2017.

[196] John W Romanishin, Kyle Gilpin, Sebastian Claici, and Daniela Rus. 3D M-Blocks: Self-Reconfiguring Robots Capable of Locomotion via Pivoting in Three Dimensions. In *The IEEE International Conference on Robotics and Automation (ICRA)*, pages 1925–1932, 2015.

[197] Kent Rosser, Jia Kok, Javaan Chahl, and Josh Bongard. Sim2real Transfer Degrades Non-Monotonically With Morphological Complexity for Flapping Wing Design. *arXiv Preprint*, 2019.

[198] Christopher Ruff, Brigitte Holt, and Erik Trinkaus. Who's Afraid of the Big Bad Wolff?: "Wolff's Law" and Bone Functional Adaptation. *American Journal of Physical Anthropology*, 129(4):484–498, 2006.

[199] Daniela Rus and Michael T Tolley. Design, Fabrication and Control of Soft Robots. *Nature*, 521(7553):467–475, 2015.

[200] Andrei A Rusu, Matej Večerík, Thomas Rothörl, Nicolas Heess, Razvan Pascanu, and Raia Hadsell. Sim-To-Real Robot Learning From Pixels With Progressive Nets. In *Conference on Robot Learning*, pages 262–270. PMLR, 2017.

[201] Tim Salimans, Jonathan Ho, Xi Chen, Szymon Sidor, and Ilya Sutskever. Evolution Strategies as a Scalable Alternative to Reinforcement Learning. *arXiv Preprint arXiv:1703.03864*, 2017.

[202] Mauro Santos, Eörs Szathmáry, and José F Fontanari. Phenotypic Plasticity, the Baldwin Effect, and the Speeding Up of Evolution: The Computational Roots of an Illusion. *Journal of Theoretical Biology*, 371:127–136, 2015.

[203] Yoshiki Sasai, Mototsugu Eiraku, and Hidetaka Suga. In Vitro Organogenesis in Three Dimensions: Self-Organising Stem Cells. *Development*, 139(22):4111–4121, 2012.

[204] Michael Schmidt and Hod Lipson. Age-Fitness Pareto Optimization. In *Genetic Programming Theory and Practice VIII*, pages 129–146. Springer, 2011.

[205] Frank Sehnke, Christian Osendorfer, Thomas Rückstieß, Alex Graves, Jan Peters, and Jürgen Schmidhuber. Parameter-Exploring Policy Gradients. *Neural Networks*, 23(4):551–559, 2010.

[206] Bernhard Sendhoff and Martin Kreutz. A Model for the Dynamic Interaction Between Evolution and Learning. *Neural Processing Letters*, 10(3):181–193, 1999.

[207] Dylan Shah, Bilige Yang, Sam Kriegman, Michael Levin, Josh Bongard, and Rebecca Kramer-Bottiglio. Shape changing robots: bioinspiration, simulation, and physical realization. *Advanced Materials*, page 2002882, 2020.

[208] Dylan S Shah, Michelle C Yuen, Liana G Tilton, Ellen J Yang, and Rebecca Kramer-Bottiglio. Morphing Robots Using Robotic Skins That Sculpt Clay. *IEEE Robotics and Automation Letters*, 4(2):2204–2211, 2019.

[209] Dylan S Shah, Joshua P Powers, Liana G Tilton, Sam Kriegman, Josh Bongard, and Rebecca Kramer-Bottiglio. Gaining Environments Through Shape Change. *arXiv Preprint arXiv:2008.06397*, 2020.

[210] Robert F Shepherd, Filip Ilievski, Wonjae Choi, Stephen A Morin, Adam A Stokes, Aaron D Mazzeo, Xin Chen, Michael Wang, and George M Whitesides. Multigait Soft Robot. *Proceedings of the National Academy of Sciences*, 108 (51):20400–20403, 2011.

[211] Yoon-Sik Shim and Chang-Hun Kim. Evolving Physically Simulated Flying Creatures for Efficient Cruising. *Artificial Life*, 12(4):561–591, 2006.

[212] Tal Shomrat and Michael Levin. An Automated Training Paradigm Reveals Long-Term Memory in Planaria and Its Persistence Through Head Regeneration. *Journal of Experimental Biology*, pages jeb–087809, 2013.

[213] George Gaylord Simpson. The Baldwin Effect. *Evolution*, 7(2):110–117, 1953.

[214] Karl Sims. Evolving 3D Morphology and Behavior by Competition. *Artificial Life*, 1(4):353–372, 1994.

[215] Karl Sims. Evolving Virtual Creatures. In *Proceedings of the 21st Annual Conference on Computer Graphics and Interactive Techniques*, pages 15–22. ACM, 1994.

[216] Hazel L Sive, Robert M Grainger, and Richard M Harland. *Early Development of Xenopus Laevis: A Laboratory Manual*. CSHL Press, 2000.

[217] EJ Slijper. Biologic Anatomical Investigations on the Bipedal Gait and Upright Posture in Mammals, With Special Reference to a Little Goat, Born Without Forelegs. In *Proceedings of the Koninklijke Nederlandse Akademie Van Wetenschappen*, volume 45, pages 288–295, 407–415, 1942.

[218] Emilie C Snell-Rood. Selective Processes in Development: Implications for the Costs and Benefits of Phenotypic Plasticity. *Integrative and Comparative Biology*, 52(1), 2012.

[219] Emilie C Snell-Rood. An Overview of the Evolutionary Causes and Consequences of Behavioural Plasticity. *Animal Behaviour*, 85(5):1004–1011, 2013.

[220] Richard Socher, Milind Ganjoo, Christopher D Manning, and Andrew Ng. Zero-Shot Learning Through Cross-Modal Transfer. In *Advances in Neural Information Processing Systems (NIPS)*, pages 935–943, 2013.

[221] Kenneth O Stanley. Compositional Pattern Producing Networks: A Novel Abstraction of Development. *Genetic Programming and Evolvable Machines*, 8(2): 131–162, 2007.

[222] Kenneth O Stanley and Risto Miikkulainen. Evolving Neural Networks Through Augmenting Topologies. *Evolutionary Computation*, 10(2):99–127, 2002.

[223] Kenneth O Stanley, David B D'Ambrosio, and Jason Gauci. A Hypercube-Based Encoding for Evolving Large-Scale Neural Networks. *Artificial Life*, 15 (2):185–212, 2009.

[224] Erik Steltz, Annan Mozeika, Nick Rodenberg, Eric Brown, and Heinrich M Jaeger. Jsel: Jamming Skin Enabled Locomotion. In *The IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5672–5677, 2009.

[225] Kelly G Sullivan, Maya Emmons-Bell, and Michael Levin. Physiological Inputs Regulate Species-Specific Anatomy During Embryogenesis and Regeneration. *Communicative & Integrative Biology*, 9(4):e1192733, 2016.

[226] Sonia E Sultan. Phenotypic Plasticity for Plant Development, Function and Life History. *Trends in Plant Science*, 5(12):537–542, 2000.

[227] Richard S Sutton and Andrew G Barto. *Reinforcement Learning: An Introduction*. MIT press, 2018.

[228] Reiji Suzuki and Takaya Arita. Interactions Between Learning and Evolution: The Outstanding Strategy Generated by the Baldwin Effect. *Biosystems*, 77 (1-3):57–71, 2004.

[229] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing Properties of Neural Networks. *arXiv Preprint arXiv:1312.6199*, 2013.

[230] Jie Tan, Tingnan Zhang, Erwin Coumans, Atil Iscen, Yunfei Bai, Danijar Hafner, Steven Bohez, and Vincent Vanhoucke. Sim-To-Real: Learning Agile Locomotion for Quadruped Robots. In *Proceedings of Robotics: Science and Systems*, 2018.

[231] Min D Tang-Schomer, James D White, Lee W Tien, L Ian Schmitt, Thomas M Valentin, Daniel J Graziano, Amy M Hopkins, Fiorenzo G Omenetto, Philip G Haydon, and David L Kaplan. Bioengineered Functional Brain-Like Cortical Tissue. *Proceedings of the National Academy of Sciences*, 111(38):13811–13816, 2014.

[232] Adrian Thompson. An Evolved Circuit, Intrinsic in Silicon, Entwined With Physics. In *International Conference on Evolvable Systems*, pages 390–405. Springer, 1996.

[233] Sebastian Thrun and Tom M Mitchell. Lifelong Robot Learning. *Robotics and Autonomous Systems*, 15(1-2):25–46, 1995.

[234] Niko Tinbergen. On Aims and Methods of Ethology. *Zeitschrift für Tierpsychologie*, 20(4):410–433, 1963.

[235] Josh Tobin, Rachel Fong, Alex Ray, Jonas Schneider, Wojciech Zaremba, and Pieter Abbeel. Domain Randomization for Transferring Deep Neural Networks From Simulation to the Real World. In *The IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2017.

[236] Josh Tobin, Lukas Biewald, Rocky Duan, Marcin Andrychowicz, Ankur Handa, Vikash Kumar, Bob McGrew, Alex Ray, Jonas Schneider, Peter Welinder, et al. Domain Randomization and Generative Models for Robotic Grasping. In *The IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3482–3489, 2018.

[237] Satoshi Toda, Lucas R Blauch, Sindy KY Tang, Leonardo Morsut, and Wendell A Lim. Programming Self-Organizing Multicellular Structures With Synthetic Cell-Cell Signaling. *Science*, 361(6398):156–162, 2018.

[238] Graham Todd, Madhavun Candadai, and Eduardo J Izquierdo. Interaction Between Evolution and Learning in Nk Fitness Landscapes. In *Artificial Life Conference Proceedings*, pages 761–767. MIT Press, 2020.

[239] Laura N Vandenberg, Dany S Adams, and Michael Levin. Normalized Shape and Location of Perturbed Craniofacial Structures in the Xenopus Tadpole Reveal an Innate Ability to Achieve Correct Morphology. *Developmental Dynamics*, 241(5):863–878, 2012.

[240] Eric D Vaughan, Ezequiel Di Paolo, and Inman R Harvey. The Evolution of Control and Adaptation in a 3D Powered Passive Dynamic Walker. In *Proceedings of the Ninth International Conference on the Simulation and Synthesis of Living Systems, Artificial Life IX*, pages 139–145. MIT Press, 2004.

[241] Roby Velez and Jeff Clune. Diffusion-Based Neuromodulation Can Eliminate Catastrophic Forgetting in Simple Neural Networks. *PloS One*, 12(11): e0187736, 2017.

[242] Jeffrey Ventrella. Explorations in the Emergence of Morphology and Locomotion Behavior in Animated Characters. In *Artificial Life IV*, pages 436–441, 1994.

[243] Jeffrey Ventrella. Designing Emergence in Animated Artificial Life Worlds. In *Virtual Worlds*, pages 143–155. Springer, 1998.

[244] Conrad H Waddington. Canalization of Development and the Inheritance of Acquired Characters. *Nature*, 150:563–565, 1942.

[245] Günter P Wagner and Lee Altenberg. Perspective: Complex Adaptations and the Evolution of Evolvability. *Evolution*, 50(3):967–976, 1996.

[246] Christoph Waldner, Magdalena Roose, and Gerhart U Ryffel. Red Fluorescent Xenopus Laevis: A New Tool for Grafting Analysis. *BMC Developmental Biology*, 9(1):1–6, 2009.

[247] Michael Wehner, Ryan L Truby, Daniel J Fitzgerald, Bobak Mosadegh, George M Whitesides, Jennifer A Lewis, and Robert J Wood. An Integrated Design and Fabrication Strategy for Entirely Soft, Autonomous Robots. *Nature*, 536(7617):451–455, 2016.

[248] Justin Werfel, Kirstin Petersen, and Radhika Nagpal. Designing Collective Behavior in a Termite-Inspired Robot Construction Team. *Science*, 343(6172): 754–758, 2014.

[249] Mary Jane West-Eberhard. Developmental Plasticity and the Origin of Species Differences. *Proceedings of the National Academy of Sciences*, 102:6543–6549, 2005.

[250] Edward L. White, Michelle C. Yuen, Jennifer C. Case, and Rebecca K. Kramer. Low-cost, facile, and scalable manufacturing of capacitive sensors for soft systems. *Advanced Materials Technologies*, 2(9):1700072, 2017.

[251] Paul White, Victor Zykov, Josh Bongard, and Hod Lipson. Three Dimensional Stochastic Reconfiguration of Modular Robots. In *Proceedings of Robotics: Science and Systems*, 2005.

[252] Julius Wolff. The Law of Bone Transformation. *German Medical Weekly (Deutsche Medizinische Wochenschrift)*, 19(47):1222–1224, 1893.

[253] Jianzhong Xi, Jacob J Schmidt, and Carlo D Montemagno. Self-Assembled Microdevices Driven by Muscle. *Nature Materials*, 4(2):180–184, 2005.

[254] Matthew D Zeiler and Rob Fergus. Visualizing and Understanding Convolutional Networks. In *European Conference on Computer Vision*, pages 818–833. Springer, 2014.

[255] Huichan Zhao, Yan Li, Ahmed Elsamadisi, and Robert Shepherd. Scalable Manufacturing of High Force Wearable Soft Actuators. *Extreme Mechanics Letters*, 3:89–104, 2015.

[256] Barret Zoph and Quoc V Le. Neural Architecture Search With Reinforcement Learning. *arXiv Preprint arXiv:1611.01578*, 2016.

[257] Victor Zykov, Efstathios Mytilinaios, Bryant Adams, and Hod Lipson. Self-Reproducing Machines. *Nature*, 435(7039):163–164, 2005.

[258] Victor Zykov, Andrew Chan, and Hod Lipson. Molecubes: An Open-Source Modular Robotics Kit. In *IROS Self-Reconfigurable Robotics Workshop*, 2007.