Growing and Evolving Soft Robots

John Rieffel[1] Davis Knox[1] Schuyler Smith[1]
Barry Trimmer[2]
[1]Union College, Schenectady, NY 12308
[2]Tufts University, Medford, MA 02155
rieffelj@union.edu

March 16, 2012

Abstract

Completely soft and flexible robots offer to revolutionize fields ranging from search-and-rescue to endoscopic surgery. One of the outstanding challenges in this burgeoning field is the chicken-and-egg problem of body/brain design: development of locomotion requires the pre-existence of a locomotion-capable body, and development of a location-capable-body requires the pre-existence of a locomotive gait. This problem is compounded by the high degree of coupling between the *material properties* of a soft body (such as stiffness or damping coefficients) and the effectiveness of a gait. This paper synthesizes four years of research into soft robotics, in particular describing three approaches to the co-discovery of soft robot morphology and control. In the first, muscle placement and firing patterns are co-evolved for a fixed body shape with fixed material properties. In the second, the material properties of a simulated soft body co-evolve alongside locomotive gaits, with body shape and muscle placement fixed. In the third, a developmental encoding is used to scalably "grow" elaborate soft body shapes from a small seed structure. Consideration of simulation time, as well as the challenges of physically implementing soft robots in the real world are discussed.

1 Introduction

Imagine a soft, resilient and deformable robot able to change shape and squeeze through small apertures. The idea of using such a robot for urban search and rescue holds great appeal, particularly in light of recent tragic earthquakes in China, New Zealand, and Japan. Once the domain of science fiction, soft robots are approaching reality – thanks to recent advances in engineering and material science. Unfortunately, the very properties which make soft robots so appealing also introduce significant obstacles, especially in the domains of design and control. Elasticity and deformability come at the cost of resonances and tight

dynamic coupling between components [23] – properties which are often assiduously avoided in conventional engineering approaches to robotic design. Small changes to the elasticity of a soft robot can cause unexpectedly large changes in performance.

The problems of soft robot design can be summarized with three questions:

- What should a soft robot *look like*? (*Morphology*)
- What should a soft robot be made out of? (*Material*)
- How should a robot *move*? (*Locomotion*)

Of course, these are not independent variables: solving each problem is predicated upon, and sensitive to, the pre-existence of solutions to the corresponding problems. The design of a soft robot's locomotive gait, for instance, depends upon both its morphology and properties such as elasticity and friction. This is in a sense an elaboration on the chicken-and-egg problem posed by body/brain design in more conventional robots [12, 13], with the added complexities which come from the effects of material properties upon a soft body's dynamics. In light of that, our approach to solving the problem will be similar: co-evolution.

The choice of a stochastic search technique like co-evolution is justified in this case given both the enormous scope of the problem and the absence of the purely analytical design methodologies available in conventional robotics. The field of evolutionary design has had considerable success in other complex design domains ranging from satellite antennae [11] to telescope lenses [1].

The ability of GAs to develop unexpectedly novel solutions to design problems is best demonstrated with an early example from our research. A small lozenge shaped robot with fixed body shape and fixed muscle placement was imported into our simulation environment, and muscle firing patterns evolved to produce locomotion. Rather than arriving at the anticipated forward-crawling motion along the long body axis, the GA instead discovered that a sideways motion (as illustrated by the frames in Figure 1) was much more efficient. (All movies referenced in this paper can be seen on YouTube by searching for the "alife-journal-softbot" tag.)

The research described in this paper focuses on three evolutionary approaches to the problem of soft robot design and control. In each case we allow two properties to vary while the others remain fixed.

> **Gait-Muscle**: Given a fixed body shape, co-evolving muscle placement and firing patterns.
>
> **Gait-Material**: Given a fixed morphology (shape and muscle placement) co-evolving material properties alongside gaits
>
> **Developmental**: Given a fixed gait, evolving open-ended and variable morphologies (implicitly varying muscle placement and material properties)

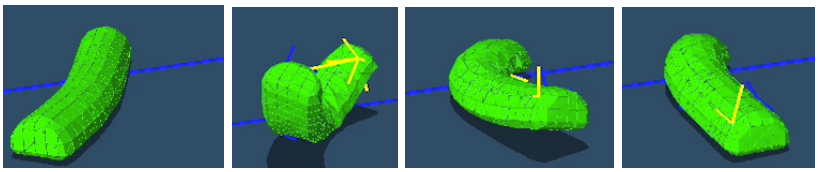

Figure 1: Genetic Algorithms often arrive at unexpectedly novel results. When asked to develop a simple locomotive gait for this lozenge-shaped robot, the algorithm arrived at a "side-winding" motion rather than the expected forward motion.

Scheme	**Gait-Muscle**	**Gait-Material**	**Developmental**
Body Shape	Fixed	Fixed	Evolved
Muscle Placement	Evolved	Fixed	Evolved
Material Properties	Fixed	Evolved	Evolved
Gait	Evolved	Evolved	Fixed

In the case of the first inquiry (**Gait-Muscle**), we will show how properties like bilateral symmetry and oppositional muscles can emerge as the result of evolution. In case of the second approach (**Gait-Material**) we will demonstrate a sensitivity between *material properties* and gaits, and arrive at unexpected forms of locomotion. In the third approach (**Developmental**) we will show how a developmental encoding can implicitly affect both material properties and muscle placement, and arrive at novel and surprising open-ended morphologies.

Salient to each approach is the problem of computational overhead imposed by simulating soft rather than rigid objects. Conventionally, soft bodies simulators ranging from Finite Element Analysis (FEA) and Computational Fluid Dynamics (CFD) to the off-the shelf physics engine used in our research (NVidia's PhysX) rely upon tetrahedral meshes to represent soft bodies, and the complexity of simulation is directly tied to the number tetrahedra in the mesh. For the fixed-morphology approach (**Gait-Material**), we introduce a method which scales model mesh resolution over the course of evolution, such that a large early portion of evolutionary time is devoted to low resolution models of the robot, and as evolution progresses mesh resolution increases. This resolution scaling achieves fitnesses comparable to those achieved by fixed high resolution while reducing overall computation time. For the evolved-morphology approach (**Developmental**), we slowly increase the developmental "lifetime" of the soft robot over the course of evolution, thereby allowing complexity of the body to increase in complexity over time.

Each exploration not only provides insight into the creation of physically grounded soft robots, but provides feedback into fields such as biomedicine and biomechanics.

2 Simulating Soft Robots

Once the domain of Finite Element Analysis (FEA) and Computational Fluid Dynamics (CFD), physics simulation is now much more accessible thanks to recent advances in commercial off-the-shelf video-game physics engines accelerated by massively parallel graphics cards (GPUs). This General Purpose Computing on Graphics Processing Units (GPGPU) can provide speedups of several orders of magnitude over software-only simulation. [2]. In particular, our research uses NVidia's PhysX engine because of its the ability to simulate complex three-dimensional *soft bodies*.

2.1 Representing Soft Bodies

In PhysX (as well as in FEA and CFD), soft bodies such as the caterpillar robots shown in Figure 3 are formed out of tetrahedral meshes . A single tetrahedron is defined by four vertices and four corresponding faces, as illustrated by figure 2. A mesh is formed by connecting adjacent tetrahedra at common vertices.

2.1.1 Soft Body Material Properties

The material properties of a soft body mesh can be tuned by varying a set of constraints placed upon the tetrahedra within a mesh. Two values, stretching stiffness and damping co-efficient, tune the parameters of a spring-and-damper system along each edge of the tetrahedron. A tetrahedral mesh with high stretching stiffness will try hardest to maintain its shape, while one with a low stiffness will flop to the floor like a deflating balloon. The damping coefficient of a soft body changes how fast it returns to equilibrium after a perturbation. A low damping co-efficient allows soft bodies in motion to oscillate more. A third constraint, volume stiffness, determines how hard each tetrahedra attempts to maintain a constant volume. Changing each of these values affects the softness of the all tetrahedra in a soft body, although not necessarily in a linear manner. As illustrated by Figure 3, by varying these material properties, the behavior of soft bodies in PhysX can range from a near fluid, to rubbery Jell-O to a semi-rigid plastic. Finally, the friction of the crawling surface can change within a relatively narrow range in order to model how well the soft material grips the substrate.

The bottleneck for soft bodies simulation is the density of the underlying tetrahedral mesh: simulation slows dramatically as the number of tetrahedra in a mesh grow (Figure 3). The trade-off is that low-resolution meshes have lower fidelity to the real-world behavior of the corresponding physical soft body.

2.2 Soft Body Gaits

One of the more interesting consequences of soft robotics is the lack of conventional actuators. Because suppleness and deformability are important, devices like servos and stepper motors are not viable. Absent those, one valuable alternative is nitinol "memory wire" [23]. These artificial muscles act essentially as linear actuators, and can be modeled in PhysX by applying equal and opposite force vectors to two attachment vertices. Figure 5 shows an example layout of the muscles on a typical soft robot.

Once given a fixed set of muscles in a soft robot, their firing patterns can be modeled with a square wave characterized by a duty cycle, a phase offset, and a period (Figure 4). The period of the firing pattern represents the time between the square wave's rising edges. Duty cycle corresponds to the percent of time that a muscle is "on" during that period. Finally, the phase of the firing pattern represents the delay before the first rising edge.

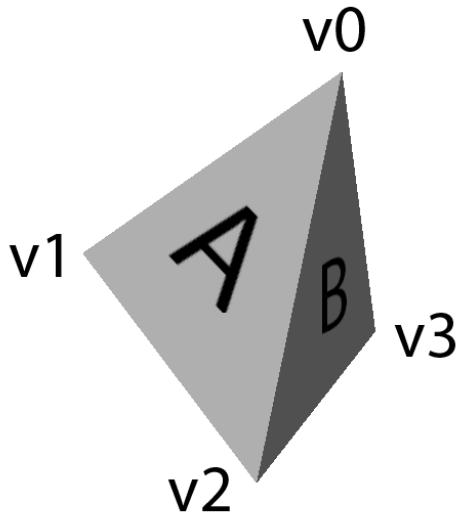

Figure 2: . A tetrahedron is defined by four vertices and four corresponding faces. The material properties of a mesh can be tuned by changing the stretching and damping coefficients of spring-and-dampers systems along the edge, and by changing the tetrahedron's resistance to volume changes.

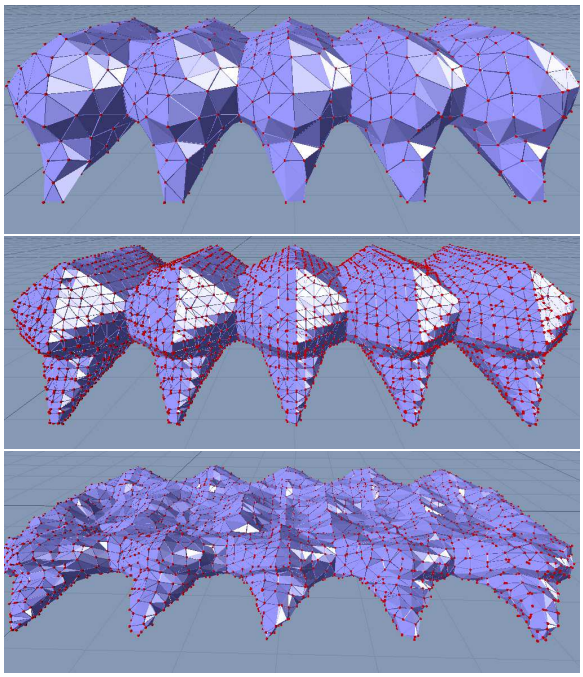

Figure 3: Soft bodies in PhysX, such as the ones shown above, are built out of meshes of tetrahedra. Model resolution can vary with the number of tetrahedra in the mesh. Example with low mesh resolution (top) and high resolution (middle). Changing the underlying material properties can drastically affect both the shape and the behavior of a soft body. Images of the same soft body with high (middle) and low (bottom) stretching stiffnesses.

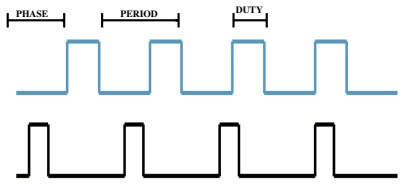

Figure 4: Soft robot gaits are composed of firing patterns for a set eight symmetrical muscles (four per side). Each of the eight patterns is described by a unique duty cycle, phase, and period.

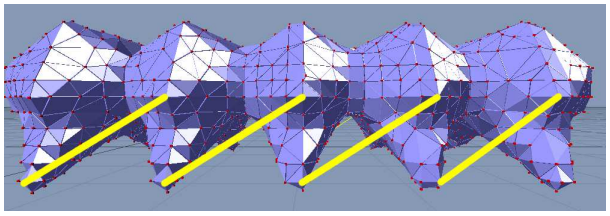

Figure 5: An illustration of the linear actuator "muscles" of the simulated soft body. Although muscles are aligned with bilateral symmetry, no symmetry constraints were placed on the underlying firing patterns.

3 Co-Evolving Muscle Placement and Gaits

One of the more fundamental choices in designing soft robots is in picking the placement of linear actuators. In our early design of soft robots [23] there were two assumptions that we took for granted, both influenced by biomimicry. The first is that muscle placement should be bilaterally symmetric, that is that the left and right side of the robot should be identical. The second is that, given the elastic nature of soft bodies, muscles acting in opposition were unnecessary – that the elastic pre-stress of the body wall would restore body shape when the linear actuator was relaxed.

In order to challenge both of these assumptions by o-evolving *muscle placement* alongside *gait* (muscle firing patterns), with no constraint placed upon symmetry or opposition. while keeping body shape and material properties fixed. The robot model we used was the caterpillar robot seen on the top of Figures 3 with 10 arbitrarily placed muscles.

The firing pattern of a single muscle was encoded in a 3-tuple of $(period, phase, duty)$ and a genome for all 10 muscles encoded as 10 such 3-tuples. A single muscle's attachment vertices were represented as a pair (v_1, v_2), and a corresponding genome contained 10 such pairs. Each linearly encoded type of genome was subject to normal per-locus mutation and two-point crossover.

The general scheme we used, as illustrated by Figure 6, was to have two parallel populations, one of gait genotypes and one of vertex genotypes. Evolution first progressed on the gait genotypes, using a hand-picked initial set of attachment vertices. After 200 generations, the current-best gait was then used as a fixed reference with which to evolve the population of attachment points. After another 200 generation interval, the current-best attachment points were used to evolve a new set of gaits. This see-saw pattern was repeated until 2000 generations of each population had been evolved.

In each population, fitness was determined by the linear distance traveled by the soft body over a fixed number (8000) of simulator time steps.

A representative solution is shown in Figure 7 (a video exists on our YouTube page). We can make two interesting qualitative observations about this evolved muscle placement. The first is that this solution, like most in our experiment, has a high degree of bilateral symmetry: most muscles have a matching muscle on the opposite side.

The second, more surprising result is the emergence of muscles acting in opposition: in this case the middle leg on the lower image is pulled both forward and backwards by a matched pair of muscles. While not strictly necessary, since the elastic force of the soft body will pull the leg back into place, an oppositional muscle is able to move the leg forward more quickly and thereby speed up the gait

While largely qualitative, these results show how relaxing otherwise assumed design constraints allows genetic algorithms to both confirm those assumptions (visavis symmetry) and improve solutions by contradicting them (visavis opposition).

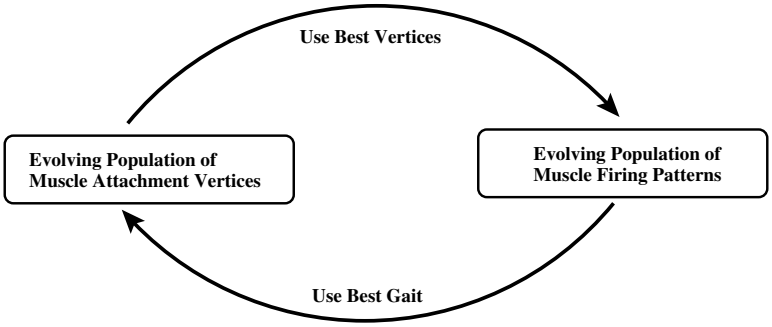

Figure 6: Co-evolving muscle placement and gaits involves two parallel populations, a population of attachment vertices and a population of muscle firing patterns. First, gaits are evolved using the current best attachment points, then 200 generations later, muscle attachment points are evolved using the current best gait.

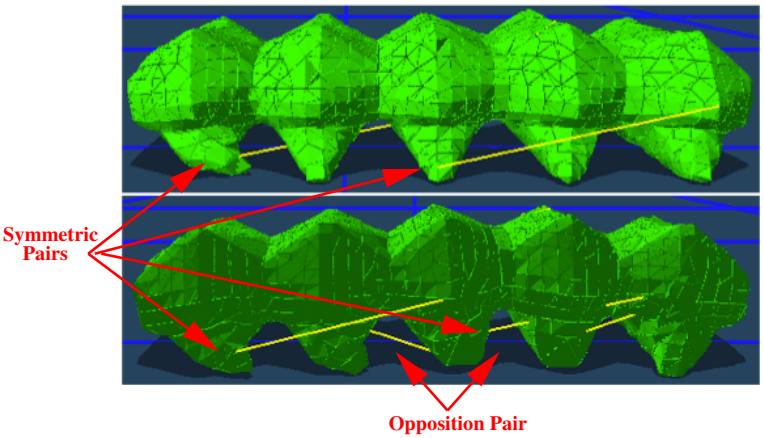

Figure 7: Images of the left and right sides of a robot with evolved muscle placement and gaits (**Gait-Muscle**). The bottom image is mirrored to provide a more direct comparison (in both images the "front" of the robot is to the right.) When muscle placement is co-evolved alongside gaits, bilateral symmetry emerges, even though it is not explicitly constrained. More unexpectedly, some muscles are placed in oppositional pairs.

4 Co-Evolving Gaits and Material

The next question we ask is, whether given a specific body plan with specific muscle placements, we can co-evolve firing patterns alongside material properties. It is worth emphasizing that this question is grounded in real-world soft robotics applications: many physical soft robots employ silicone elastomers, whose material properties can be changed quite significantly during the mixing process [23].

Our goal in this case was to simultaneously discover a suitably matched gait/property pair. Because of the dynamics of a soft body in motion the fitness of a specific gait can vary greatly depending upon the underlying material properties, and, similarly, the fitness of a material property set depends greatly upon the gait it is tested against. Our co-evolutionary scheme in this case is identical to that in Section 3, except that while one population contained gait genomes for eight muscles, the other contained genomes representing material properties (as described in Section 2.1.1). Property values were limited to keep results realistic. Ranges are as follows (note that in PhysX, like most physics simulators, these properties are unit-less):

Property	Min	Max
Volume Stiffness	0.1	1.0
Stretching Stiffness	0.3	1.0
Friction	0.5	1.0
Damping	0.0	1.0

Initially, a fixed "best guess" of material property values was used for evaluating the fitness of each gait. The second population evolved soft body material properties, where a single genome contained values for a specific set of stretching stiffness, volume stiffness, damping co-efficient and body friction. Initially, for this population's fitness evaluations, a fixed "best guess" of firing patterns was used.

Every tenth generation, the gait used for material property fitness evaluation was updated with the current highest-fitness gait from the gait population, and the material properties used for gait population evaluation were updated from the highest-fitness property values.

As a measurement of "wall time", with population size 40, a typical run took 24 hours to evaluate 100 generations of each population on a 2.66 GHz Core i7 processor with 6 GB of RAM.

We have two analyses of our experiments to offer. The first is a more qualitative description of the gaits produced by our system and some insight into how changing material properties affect fitness. The second more quantitatively explores the effect of scaling the mesh resolution over the course of an evolutionary run.

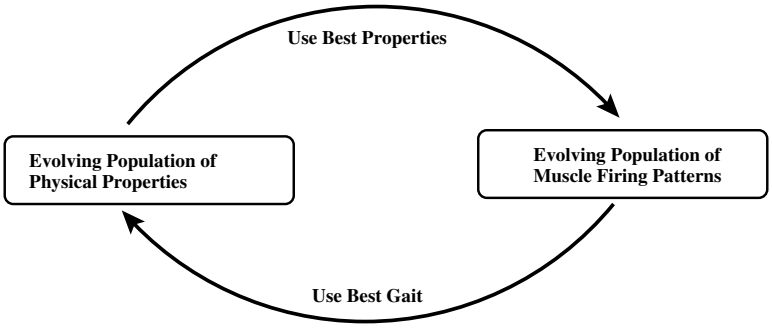

Figure 8: Similarly, Co-evolving gaits with physical properties requires a population of gaits and a population of physical properties.

4.1 Analyzing effects of Material Properties on Gaits

Our experiments consistently produced interesting and effective gaits, and analysis suggests that being able to change material properties alongside firing patterns has a positive effect upon the outcome. A qualitative and visual representation lies in videos of the actual gaits available on our YouTube channel.

One video shows a bipedal - that is, bilaterally asymmetric – gait. Firing patterns on each side of the body co-ordinate in a rough front-to-back wave pattern in order to collectively lift the limbs upwards and forwards during the upswing, before relaxing into the downswing to pull against the ground. The relative softness of the material can be seen in the amount of flexing undergone by each leg. A second video by comparison shows a more symmetrical gait achieved by a forward-moving wave which produces what almost looks like a gallop.

There were some distinct differences in material property values across these two runs, as summarized below:

Property	Bipedal	Wave
Volume Stiffness	0.986	0.996
Stretching Stiffness	0.982	0.998
Friction	0.598	0.804
Damping	0.0004	0.0

The most notable difference is the friction – corresponding to the stickiness of the robot's feet, however when watching the videos, the relatively minor numerical differences in the other property values appear, at least qualitatively, to be reflected in the behavior of the soft bodies.

Of further interest is the change in best-of material values properties which occur over the course of an evolutionary run, as shown in Figure 10. While damping coefficient and volume stiffness show relatively monotonic progress toward a fixed value, stretching stiffness and friction vary consistently across a relatively wide range during evolution. The effect of material property changes on fitness is even more apparent when shown alongside the corresponding fitness graph (shown on the bottom of Figure 10). The large swing in damping coefficient at generation 7 corresponds to a matching significant rise in fitness. Other, smaller, fitness gains also appear to have corresponding material value changes.

4.2 Scaling Mesh Resolution

Our second analysis is of the benefits offered by scaling mesh resolution over the course of evolution. Recall that the number of tetrahedra in a mesh are the determining factor in simulation run time, as well as in simulator fidelity. We ran a suite of experiments exploring the effects of different scaling schemes, as summarized in Table 1.

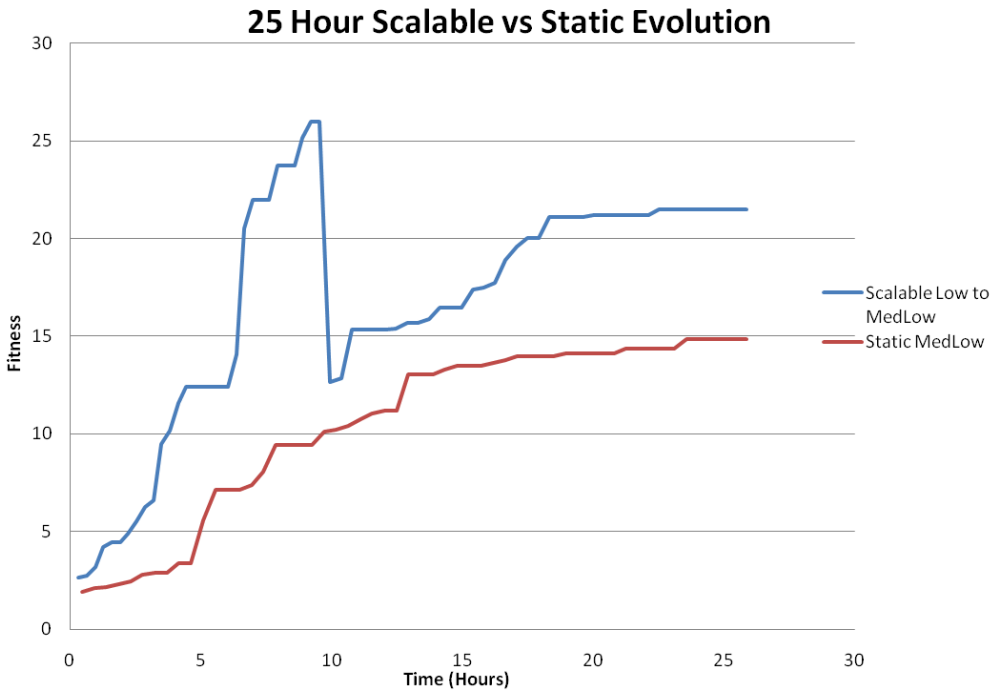

Figure 9: A comparison of co-evolutionary progress on a static mesh resolution (bottom) vs. a single resolution switch. The sudden drop in fitness corresponding to resolution change is caused by the low-resolution gait working less well on the higher resolution mesh.

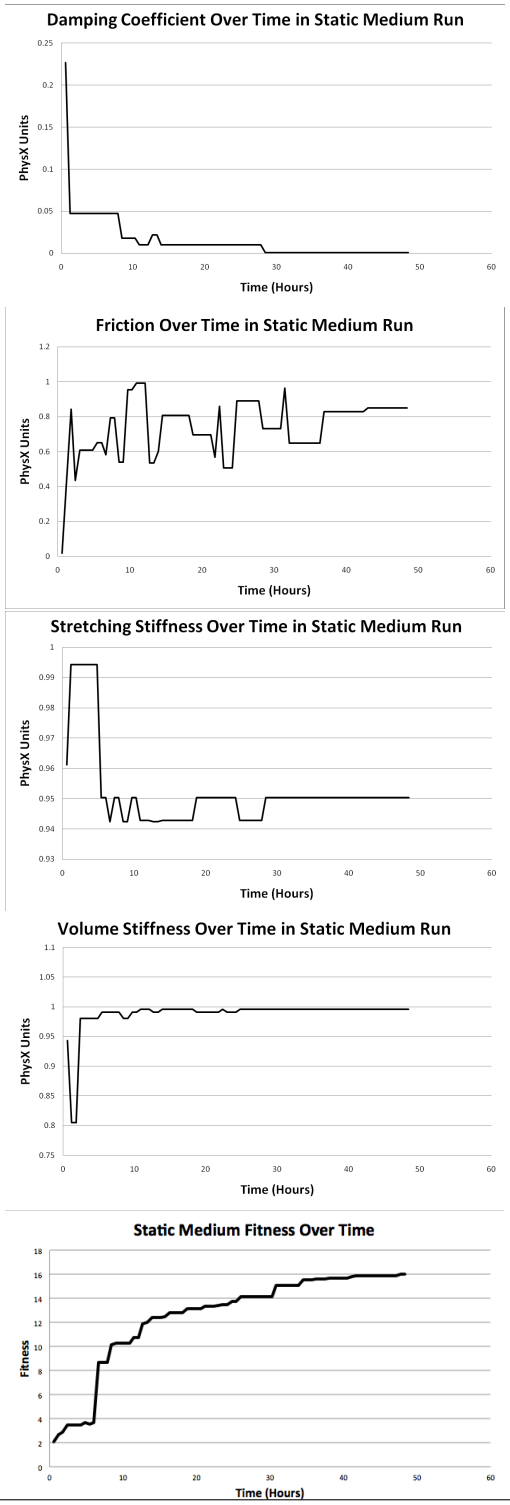

Figure 10: Material property values for population bests per generation. Large swings in values, such as the drop in damping at generation 7, are co-ordinated with large improvements in fitness (bottom figure)

Our intuition was that the the bulk of early evolutionary time, which largely consist of the soft robot flailing around – that is attempting to achieve non-zero fitness, could be performed on relatively low meshes, and then as evolution and fitness progressed, mesh resolution could be scaled upwards to raise the emphasis on fidelity at the cost of longer evaluation times.

There were five mesh resolutions available to the system: low, medium low, med, high, and maximum. Runs could switch 0,1,2 or 4 times. All of the non-static runs shown began on the low mesh – runs that are listed with a mesh switch count of one, for instance, changed from the low mesh to their end mesh. Runs listed with a mesh switch count greater than one ran on an intermediate mesh(es) before reaching their end mesh. All other properties, such as population size, remained constant across experiments. Resolution changes occurred every 30 generations.

Fitness	Hours	End Mesh	Mesh Switches
40.16	34	Low	0
46.94	42	Low	0
24.61	72	Max	0
14.86	25	MedLow	0
21.47	25	MedLow	1
11.97	24	Med	0
12.86	24	Med	1
15.99	48	Med	0
16.65	48	Med	1
15.91	48	Med	2
6.18	46	Max	4

Table 1: A summary of results from resolution scaling

Figure 9, which compares evolution with a single (low-to-medium-low) switch to that with a fixed (medium-low) mesh resolution show the consequences of this process: fitness in the scalable evolution during its "low" phase progresses much more rapidly during the first 10 hours of simulation. Once the phase change into a higher mesh density occurs, however, there is a dramatic drop in fitness, and the scalable run loses much of the ground it had gained (though it still remains above the fixed resolution result). During the following 15 hours, the scalable run is able to make up much of the lost fitness, and improves more rapidly than the static mesh.

This steep loss in fitness is due to the large extent to which the success of a gait is highly tuned to its specific mesh resolution. The same actually holds true of the evolving material properties as well. Gaits and physical properties evolved at one mesh resolution simply do not translate perfectly when placed in a higher resolution simulation.

This dependence on mesh resolution also has a clear effect upon the the maximum obtainable evolutionary fitness: over similar time scales, even the static meshes show significant differences in final fitness.

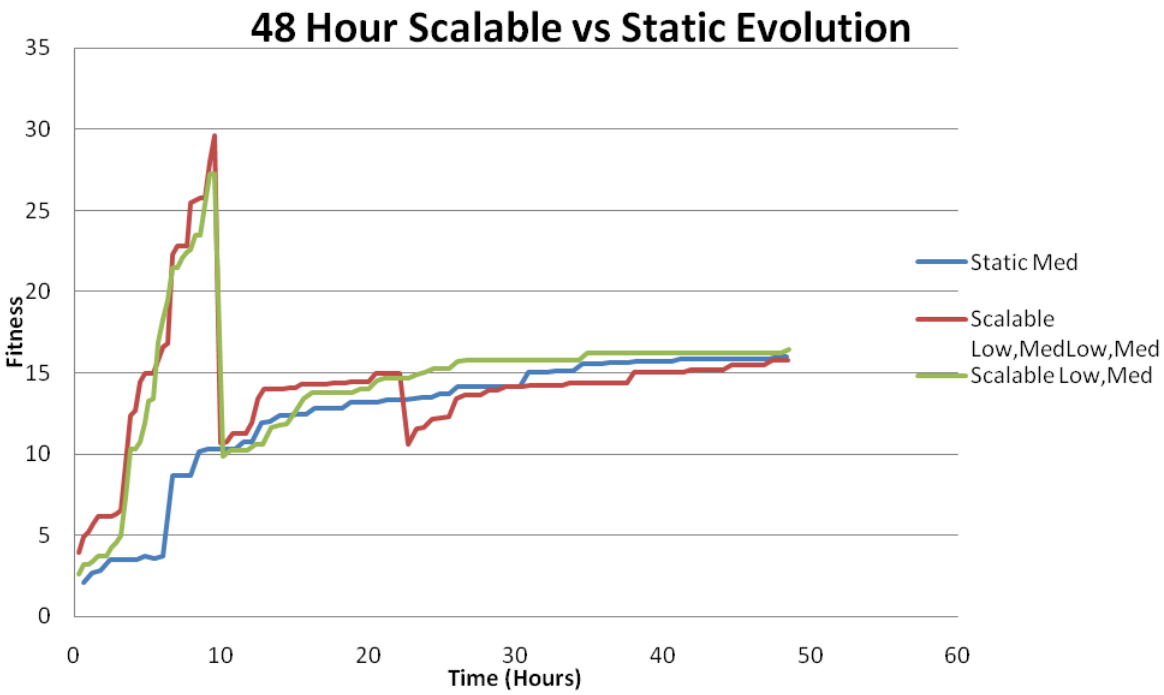

Figure 11: Results comparing the consequence of switching mesh resolution multiple times over the course of evolution. Each time resolution changes there is a dramatic drop in fitness due to the relatively poor translation of gaits and material properties into the higher mesh.

The last entry in Table 1 illustrates the cost of switching clearly: the final fitness is less than half of that achieved by any other run. This suggests that, in its current form, sometimes the cost of resolution scaling can be too high. Figure 11 shows a case where even a single switch in resolution results in an equivocal, at best, improvement in overall fitness.

The source of this loss in fitness can possibly be illustrated with an interesting qualitative distinction of gaits evolved at varying resolutions: all the gaits evolved in a low resolution mesh produced bi-pedal gaits, whereas gaits produced in the "maximum" mesh tended to be more bilaterally symmetric, involving instead a forward-propagating wave-like motion. In other words, sauce for the (low-resolution) goose may not be sauce for the (high-resolution) gander. A high-fitness bipedal gait evolved a low mesh resolution ceases to be competitive when placed in a higher-resolution body.

Mesh scaling certainly holds promise, and in a few cases illustrated above, offers an improvement over static-resolution evolution, despite the large fitness drops associated with resolution switches. While it remains to be seen if this is a viable way to address the issue of long simulation times, we are hopeful of its prospects. Our next section explores an alternative method for variable mesh resolution.

5 Evolving Body Shape Development

In our final approach we use a grammatically-based developmental encoding to evolve complete soft robot shapes in an open-ended manner. As we'll show, this allows us to address several of the challenges of soft robotics discussed above.

Developmental encodings have been used with considerable success to create the morphology of simulated robotics [19, 4, 5, 9, 14, 10], most often in conjunction with off-the-shelf physics engines. While there are several different flavors of generative encoding (among them L-Systems [10, 15], Genetic Regulatory Networks [3], and HyperNEAT [21] – Stanley and Miikkulainen [22] provide a useful taxonomy), they are largely used to create *rigid* structures, and none explicitly operate on tetrahedral meshes.

For our purposes, the open-ended generation of tetrahedral meshes, we have created a face encoding L-system [16] capable of growing multi-resolution tetrahedral-mesh robot morphologies. L-Systems use a sequence of rewrite rules which operate on the faces of tetrahedra. Similar encodings operating on graph edges rather than tetrahedral faces have been used to grow both 3-D surfaces[6] and tensegrity structures [17].

The details of our particular encoding are as follows. Assuming that each face of a tetrahedron can be given a label, we specify three operations which can be performed upon a face, as illustrated in Figure 12. For the sake of simplicity, we assert that these operators can only be applied to *exposed* faces – that is, those which are not shared by two adjacent tetrahedra – most commonly, but not exclusively, on the exposed outer surface of a tetrahedral mesh. A "ruleset" can then be created by specifying a fixed number of nonterminal labels, and

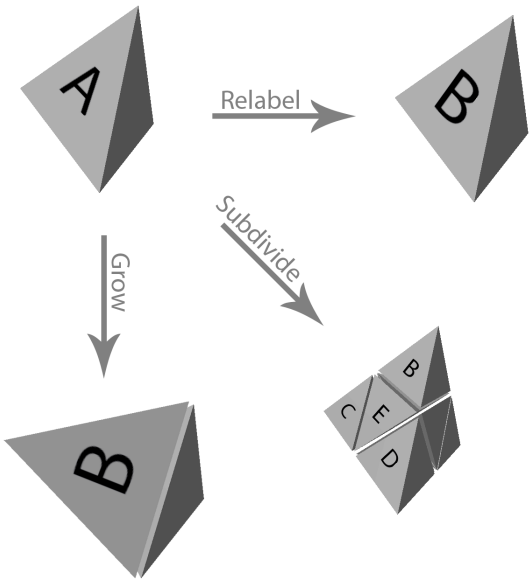

Figure 12: An illustration of the three rules which can be applied to the face of a tetrahedron. Clockwise from top left: the original tetrahedron with face labeled "A", **relabel** replaces "A" with "B", **subdivide** replaces the face with four smaller faces (this requires subdividing the entire tetrahedron), and **grow** adds a new tetrahedron with face labels "B","C","D"

providing a rewrite rule for each label. A detailed description of this encoding is provided by an earlier paper of ours [16].

Tetrahedral meshes of arbitrary size can now be grown by iteratively applying a ruleset to an initial "seed" tetrahedron like the one shown in Figure 2. Each exposed face of the growing tetrahedral mesh is kept in a queue, and is associated with three vertices and exactly one tetrahedron. (A face shared by two tetrahedra is by definition not exposed.) For every generation of growth, the open faces are iteratively removed from the queue and the appropriate rule for their is applied. For *relabel*, a new face with the new label is enqueued. For *grow* and *divide*, new vertices and tetrahedra are computed and added, and

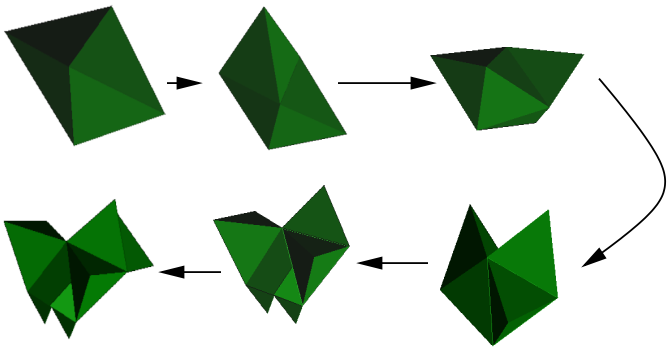

Figure 13: The growth of a tetrahedral mesh by iteratively applying the rules from a face encoding ruleset.

then the resulting three (*grow*) or four (*subdivided*) new faces are enqueued. Figure 13 shows the growth of one such tetrahedral mesh.

5.1 Implicitly Variable Mesh resolution

An valuable consequence of this encoding, particularly in light of our efforts in Section 4.2 is that it can result in tetrahedral meshes of *variable resolution*: each *subdivide* operation on a face converts the single parent tetrahedron into eight smaller tetrahedra, any of which can in turn be subdivided into eight smaller tetrahedra 1/64th the size of the original tetrahedron. Meanwhile, any tetrahedron not subject to face subdivisions will retains its original dimensions. Since the time complexity of a soft body simulation is determined largely by the total number of tetrahedra, this variable resolution allows for a certain reduction in simulation time compared to an identical tetrahedral mesh composed of homogeneously sized tetrahedra. As we will see in the upcoming section, this has consequences for differential stiffness within the mesh as well.

Rulesets describe a developmental *process*, and as such they are an *indirect encoding* of a tetrahedral mesh physical phenotype. These grammars can be evolved by treating the rulesets as genotypes. and the tetrahedral mesh which results after a fixed number of iterations as the phenotype. With a grammar as a genotype, an evolving population simply consists of collection of these grammar genotypes.

5.2 Moving Without Linear Actuators

Using linear actuators with attachment vertices as in our previous two approaches would have further complicated the grammar. We chose instead to periodically vary the stiffness of the tetrahedral mesh, which results in corre-

A	$\rightarrow$	grow	$\{DBF\}$
B	$\rightarrow$	grow	$\{ADF\}$
C	$\rightarrow$	grow	$\{EDF\}$
D	$\rightarrow$	relabel	(D)
E	$\rightarrow$	grow	$\{DCF\}$
F	$\rightarrow$	divide	$[DDDG]$
G	$\rightarrow$	grow	$\{DDG\}$

Table 2: An example ruleset.

sponding deformations in the soft body itself. (Hiller and Lipson used a similar approach in their amorphous robots [7], which they were able to physically implement using closed cell foam and a vacuum chamber [8].) It is worth noting that in this PhysX implementation a uniform change in stiffness can result in non-uniform deformations, since a region of smaller tetrahedra will deform more than region with a single large tetrahedron when under similar loads, largely due to the increased number of vertices around which the tetrahedra can flex.

5.3 Evolving Morphology with Grammars

Rulesets such as the one shown in Table 2 describe a developmental *process*, or *ontogeny*. As such they are an *indirect encoding* of a physical phenotype. These grammars can be evolved by treating the rulesets as genotypes, and the tetrahedral mesh which results after a fixed number of iterations as the phenotype. Rulesets can be encoded as simple linear strings of characters subject to mutation and crossover. Mutation on a grammar genotype affects only the right hand side of a rule, and can either change the rule (i.e. $grow(A, B, D) \rightarrow relabel(A)$, with extra labels added or removed as necessary) or change a label in the rule (i.e. $grow(A, B, D) \rightarrow grow(E, B, D)$. Single point crossover grabs a subset of production rules from one parent, and the remainder from a second. An evolving population simply consists of collection of grammar genotypes.

For the experiments which produced the results described below, we used a fixed population size of 20 and 50% elitism. Parents were chosen with a simple fitness proportional selection. 40% of offspring were produced via crossover, and the remainder via mutation. An edit-distance diversity metric was employed to prevent multiple neutral mutations of a single genotype schema to proliferate.

Each individual was evaluated by applying its genotype grammar rewrite rules a fixed number of times (20, 40, or 100) in order to produce a phenotype, and then placing the resulting robot in the PhysX environment. Tetrahedral mesh stiffness was then cycled between between maximal stiffness and 80% of maximal (again, all parameters in PhysX are unit-less), with a period of 200 time steps, and the displacement vector over each cycle recorded. Total distance traveled during a 4000 time step window was then computed by adding the displacement vectors.

In order to prevent the common but pathological trait of toppling, in which

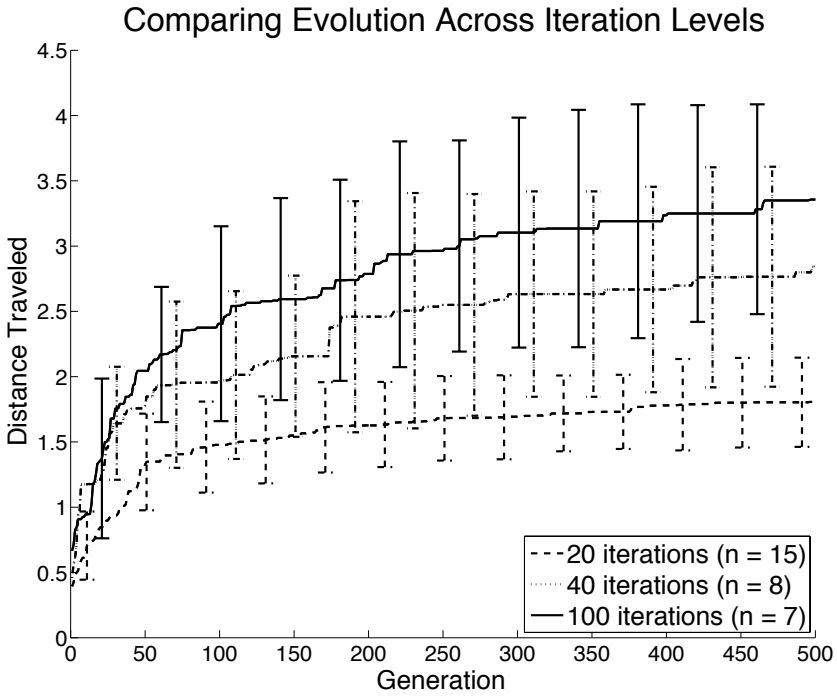

Figure 14: A comparison of evolutionary progress of tetrahedral meshes produced after 20 iterations against those produced after 40 iterations. More iterations tend to produce larger meshes.

evolving bodies move their centers of mass by simply falling over (first made famous by Sim's evolved agents [20], the displacement vectors were passed through a low-pass filter, which effectively ignores short term high velocities associated with toppling, while preserving the more steady state long term slower velocities due to actual locomotion.

In each experiment, evolution proceeded for 500 generations. Each grammar was fixed at 7 labels A through G, with seven corresponding rewrite rules.

5.4 Results and Discussion

Every experiment across both iteration levels (20, 40, 100) produced forward motion of the soft bodies, with no fitness exploitation via toppling. Videos of several of the evolved soft robots can be seen on our YouTube channel. Figure 14 compares fitness over time across the three iteration levels. As might be expected, the bodies produced with 100 iterations traveled further in the same amount of time than those generated with either 40 or 20 iterations.

Figure 15 contains frames from one such video. In almost every case, the

evolved gait involves a "scooting" or "ratcheting" motion, where cyclic flexing on one end robot pushes the robot in the opposite direction.

Figure 15: Frames from a video of the evolved locomotion. Cyclic changes in stiffness affect the "leg" on the left, and ratchets the rest of the body to the right.

Elements of body symmetry and modularity are visible in most of the evolved morphology. Symmetry is somewhat limited by the choice to hard-code the faces of the seed tetrahedron as $ABCD$ – this meant that in order to achieve two radially symmetric limbs growing from the original tetrahedron, rewrite rules which relabeled two of the starting faces to a third common face would be required - something along the lines of $A \rightarrow relabelG, B \rightarrow relabelG$. In future experiments we intend to make the initial starting labels a component of the genome, which will allow for more symmetry, and also for more large-scale co-ordinated changes in the phenotype.

5.4.1 Grammars Implicitly Specify Both Muscle Function and Material Properties

As illustrated by the results, the multi-resolution nature of the evolved tetrahedral meshes allows for a significant degree of regional functional differentiation. What we mean by this is that regions of the body composed of larger tetrahedra (lower resolution) tend to remain stiff over the course of locomotion – thereby providing structure to the body – whereas regions with many smaller tetrahedra flex more, and thereby act more like muscles. Figure 16 illustrates this phenomenon more clearly. The region of smaller tetrahedra on the left of each frame compress and buckle as their stiffness is reduced, while the larger tetrahedra pivot around the buckling. As stiffness increases again, the structure is propelled to the right.

While the grammar *explicitly* determines body shape, this regional differentiation is an emergent property of our grammar. This is quite valuable in that it allows our grammars to implicitly specify both material properties (in that regions with smaller tetrahedra are more flexible) as well as function (regions with smaller tetrahedra act like muscles). Moreover, since mesh resolution can vary heterogeneously throughout the body we are allowed a much finer-grained control over mesh resolution than the explicit resolution switching used in Section 4.

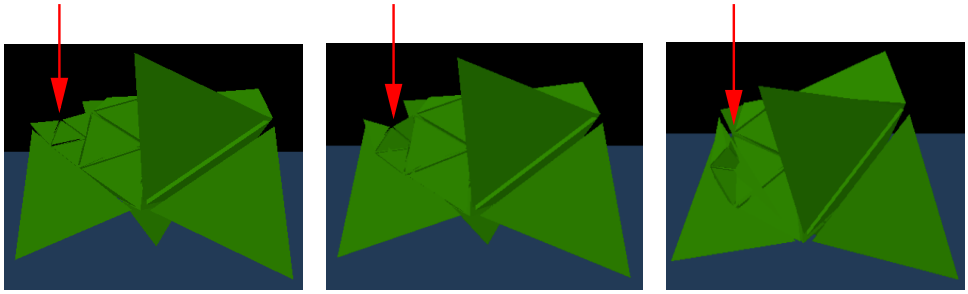

Figure 16: An illustration of the regional functional differentiation exhibited by the evolved meshes. A region with smaller tetrahedra, indicated by the red arrow, is more flexible and tends to buckle as stiffness is decreased, while larger tetrahedra are less affected.

5.5 An Avenue toward open-ended greater scalability

It is worth noting that Figure 14 is slightly misleading in the sense that this is not a purely apples-to-apples comparison. Tetrahedral meshes generated with 20 iterations generally have fewer tetrahedra than those generated with 40, or 100 iterations. Since computational complexity is proportional to the the number of tetrahedra, a "wall time" comparison of the three values might be better means of comparison.

This does, however, lead to a second observation, more quantitative in nature, which can be seen when the developmental trajectory of an evolved solution is inspected more closely. Over the course of evolution, fitness of the tetrahedral mesh is measured only once – after a predetermined number of rewrite rules have been applied. There is, however, an underlying ontogenic process, which means that every intermediate phenotype between original seed structure and "fully grown" structure is itself a complete and viable tetrahedral mesh. Given an evolved grammar, we can therefore measure the fitness of every intermediate stage of development.

Since the fitness after N rewrites is the only one that matters in our evolutionary setup, the fitnesses of prior (and subsequent) developmental stages isn't selected for, and therefore there is no expectation that those stages would have any significant fitness. However, as the graphs in Figure 17 illustrate, instead there can be a high degree of fitness throughout the developmental trajectory. Sometimes, as in the top graph, there is a sharp jump in fitness near the fixed 20-iteration limit, but fitness remains quite stable for an additional 10 iterations before slowly decaying. In other cases (middle graph), fitness builds gradually (and non-monotonically) up to the iteration limit, and then drops to around 30% of nominal fitness for the next 10 iterations, and actually jumps up above 50% of nominal fitness after 15 iterations. In a third case (lower figure), there are ontogenic stages before and after the iteration limit which are twice as high as the evolved fitness value.

This suggests a new avenue to follow for this developmental encoding: gradually scaling the number of iterations over the course of evolution. This would in effect offer a scaling process very similar to that we describe in Section 4, but without the need to hand-code mesh resolution.

5.6 Tetrahedral Robots can be Printed

Although these tetrahedral meshes were evolved in simulation, they are exceptionally easy to fabricate in the real world. A stereo lithography (STL) file, readable by all modern 3-D printers and rapid prototyping machines, can be generated from a tetrahedral mesh description simply by enumerating the open faces. Figure 18 provides an example of one such mesh which was printed out of ABS plastic on a Stratasys printer. Printed robots need not be rigid, of course: state-of-the-art printers, such as those designed by Objet, are now able to print in much softer materials, and even cheap desktop prototypers like the Makerbot

Cupcake can print 3-D shapes out of silicone elastomers.

6 Conclusion: Three Ways to Evolve Soft Robots

The evolution of completely soft robots is a many-headed problem. The three central challenges, morphology, material, and control, are all interdependent, and a solution for any one is predicated upon existing solutions to the other two.

We have explored three ways to approach this problem, in each case holding one property constant while co-evolving the other two. In the first two cases, each with fixed morphology, this required two separately evolving populations, where the current-best from one parameter's population (for instance, gait) was used as the baseline to evolve a second parameter (for instance muscle location). In the third case, we held control constant while using a developmental encoding which was able to simultaneously control both muscle placement and material properties by varying tetrahedral mesh size.

While each approach highlights the challenges involved in soft robot evolution, each leads to interesting insights as well.

Results from our first approach, **Gait-Muscle**, suggest that bilateral symmetry and oppositional muscles can be valuable in soft robotic gaits. Our second approach, **Gait-Material**, highlight the tight coupling between soft robotic material properties and gaits, and suggest that both should be evolved in tandem. This emphasis on the effects of body dynamics have a recent biological analog as well in the discovery that in the *Manduca sexta* caterpillar, the biomimetic muse of this research, the internal gut plays a surprisingly important role in locomotion [18].

We find the most promise in our third approach, **Developmental**, through which a generative encoding is able to explicitly generate body *shape* while implicitly determining heterogeneous material properties throughout the body as well as muscle placement. In our upcoming work we look forward to leveraging the developmental stability of the grammatical encoding in order to find a way to scale to increasingly large body sizes without determining the number of iterations *a priori*.

Taken collectively, we hope that these these insights can inform our efforts at achieving physically embodied soft robots. Success in this regard will have valuable applications in fields ranging from soft and flexible urban search and rescue robots to more compliant biomedical and orthoscopic devices.

References

[1] S. H. Al-Sakran, J. R. Koza, and L. W. Jones. Automated re-invention of a previously patented optical lens system using genetic programming. In M. Keijzer, A. Tettamanzi, P. Collet, J. I. van Hemert, and M. Tomassini, editors, *Proceedings of the 8th European Conference on Genetic Programming*, volume 3447 of *Lecture Notes in Computer Science*, pages 25–37, Lausanne, Switzerland, 2005. Springer.

[2] W. Banzhaf and S. Harding. Accelerating evolutionary computation with graphics processing units. In *GECCO '09: Proceedings of the 11th annual conference companion on Genetic and evolutionary computation conference*, pages 3237–3286, New York, NY, USA, 2009. ACM.

[3] J. Bongard. Evolving modular genetic regulatory networks. In *Evolutionary Computation, 2002. CEC '02. Proceedings of the 2002 Congress on*, volume 2, pages 1872 –1877, 2002.

[4] J. Bongard and R. Pfeifer. *Morpho-functional Machines: The New Species (Designing Embodied Intelligence)*, chapter Evolving complete agents using artificial ontogeny, pages 237–258. Springer-Verlag, Berlin, 2003.

[5] J. C. Bongard and R. Pfeifer. Repeated structure and dissociation of genotypic and phenotypic complexity in artificial ontogeny. In L. Spector et al., editors, *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2001)*, pages 829–836, San Francisco, California, USA, 7-11 2001. Morgan Kaufmann.

[6] M. Hemberg and U.-M. O'Reilly. Extending grammatical evolution to evolve digital surfaces with genr8. In *EuroGP*, 2004.

[7] J. D. Hiller and H. Lipson. Evolving Amorphous Robots. In H. Fellermann, M. Dörr, M. M. Hanczyc, L. L. Laursen, S. Maurer, D. Merkle, P.-A. Monnard, K. Stoy, and S. Rasmussen, editors, *Artificial Life XII: Proceedings of the Twelfth International Conference on the Simulation and Synthesis of Living Systems*, pages 717–724. MIT Press, Cambridge, MA, Aug. 2010.

[8] J. D. Hiller and H. Lipson. Morphological evolution of freeform robots. In *Proceedings of the 12th annual conference on Genetic and evolutionary computation*, GECCO '10, pages 151–152, New York, NY, USA, 2010. ACM.

[9] G. S. Hornby and J. B. Pollack. The advantages of generative grammatical encodings for physical design. In *Proceedings of the 2001 Congress on Evolutionary Computation CEC2001*, pages 600–607, COEX, World Trade Center, 159 Samseong-dong, Gangnam-gu, Seoul, Korea, 27-30 2001. IEEE Press.

[10] G. S. Hornby and J. B. Pollack. Evolving l-system to generate virtual creatures. *Computers and Graphics*, 25(6):1041–1048, 2001.

[11] J. D. Lohn, G. S. Hornby, and D. S. Linden. An Evolved Antenna for Deployment on NASA's Space Technology 5 Mission. In U.-M. O'Reilly, R. L. Riolo, T. Yu, and B. Worzel, editors, *Genetic Programming Theory and Practice II*. Kluwer, 2005.

[12] J. Pollack, H. Lipson, P. Funes, S. Ficici, and G. Hornby. Coevolutionary robotics. In *EH '99: Proceedings of the 1st NASA/DOD workshop on Evolvable Hardware*, page 208, Washington, DC, USA, 1999. IEEE Computer Society.

[13] J. B. Pollack, H. Lipson, G. Hornby, and P. Funes. Three generations of automatically designed robots. *Artificial Life*, 7(3):215–223, Summer 2001.

[14] J. B. Pollack, H. Lipson, G. Hornby, and P. Funes. Three generations of automatically designed robots. *Artifial Life*, 7(3):215–223, 2001.

[15] P. Prusinkiewicz and A. Lindenmayer. *The Algorithmic Beauty of Plants*. Springer–Verlag, New York, USA, 1990.

[16] J. Rieffel and S. Smith. A face-encoding grammar for the generation of tetrahedral-mesh soft bodies. In H. Fellermann, M. Dörr, M. M. Hanczyc, L. L. Laursen, S. Maurer, D. Merkle, P.-A. Monnard, K. Stoy, and S. Rasmussen, editors, *Artificial Life XII: Proceedings of the Twelfth International Conference on the Simulation and Synthesis of Living Systems*, pages 414–420. MIT Press, Cambridge, MA, Aug. 2010.

[17] J. Rieffel, F. Valero-Cuevas, and H. Lipson. Automated discovery and optimization of large irregular tensegrity structures. *Computers & Structures*, 87(5-6):368 – 379, 2009.

[18] M. Simon, W. Woods, Y. Serebrenik, S. Simon, L. V. Griethuijsen, J. Socha, W. Lee, and B. Trimmer. Visceral-locomotory pistoning in crawling caterpillars. *Current Biology*, 20:1458–1463, 2010.

[19] K. Sims. Evolving 3d morphology and behavior by competition. In R. Brooks and P. Maes, editors, *Artificial Life IV Proceedings*, pages 28–39. MIT Press, 1994.

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

[21] K. O. Stanley, D. B. D'Ambrosio, and J. Gauci. A hypercube-based encoding for evolving large-scale neural networks. *Artif. Life*, 15:185–212, April 2009.

[22] K. O. Stanley and R. Miikkulainen. A taxonomy for articial embryogeny. *Artificial Life*, 9(2):93–130, 2002.

[23] B. Trimmer. New challengees in biorobotics: incorporating soft tissue into control systems. In *IEEE International Conference on Robotics and Automation*, 2007.

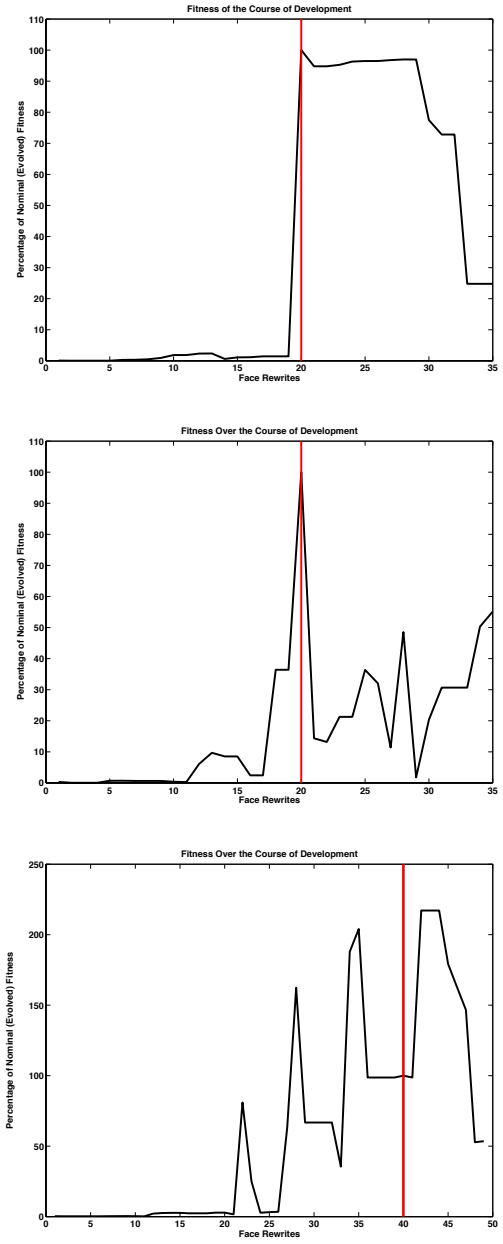

Figure 17: Phenotype fitness measured over the course of development, and normalized around the fitness measured at the fixed iteration limit (vertical red line). Although during evolution fitness is only measured at that iteration limit, fitness often remains stable as development proceeds past the limit (top graph), and fitness is sometimes higher at earlier and later developmental stages.

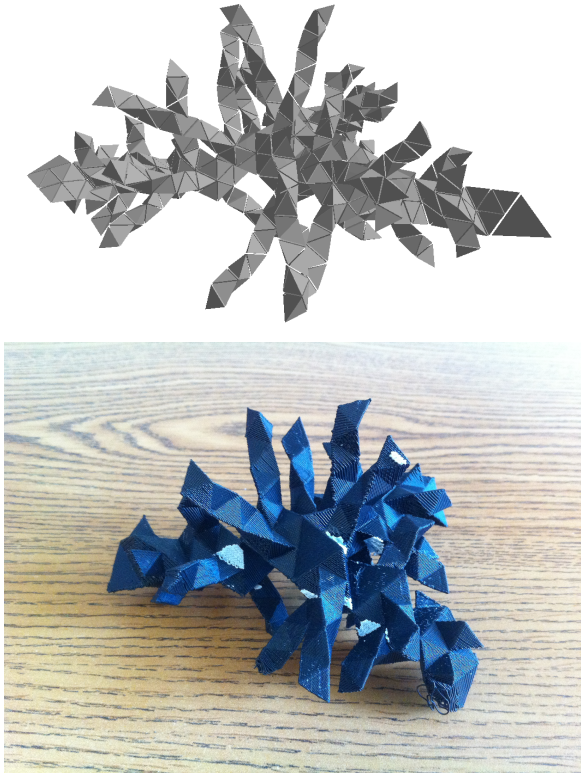

Figure 18: Tetrahedral meshes (left) can easily be converted into stereo lithography format and printed with high fidelity on a 3-D printer (right).