

Using Morphogenetic Models to Develop Spatial Structures

Jacob Beal*, Jessica Lowell*, Annan Mozeika[†] and Kyle Usbeck*

*BBN Technologies, Cambridge, Massachusetts, USA 02138

[†]iRobot Corporation, Bedford, Massachusetts, USA 01730

Email: {jakebeal, jlowell, kusbeck}@bbn.com, amozeika@irobot.com

Abstract—A common problem in spatial computing is how to arrange the structure of a spatial computer into a geometric form adapted for its current environment and needs. In natural biological organisms, the processes of morphogenesis adapt structure to environment remarkably well on both an individual and evolutionary time scale. However, no clear framework has been developed for exploiting morphogenetic principles in the creation of engineered systems. In this paper, we present preliminary work toward such a framework, developed against the example of a robot similar to the iRobot LANDroid. We first show how developmental programs might act as a reference architecture for engineered designs, facilitating variation. We then present a candidate basis set of geometric operations for encoding adaptable developmental programs, demonstrate how they can be applied to develop a robot body plan, and discuss progress toward implementation.

I. INTRODUCTION

A common problem in spatial computing is how to arrange the structure of a spatial computer into a geometric form adapted for the current circumstances of environment and program execution. For example, blueprints for distributed construction [1], must both ensure that construction can progress autonomously and that the final structure is suited for its environment. Other examples include swarm robots that collaborate on tasks too large for a single robot [2], swarm “pseudopods” that search through a building while maintaining a connection to a home base [3], actuated structures that adjust to maintain a stable surface [4], and the engineered growth of biological tissues [5]. The challenge is to find representations of geometric form that facilitate the design and adaptation of geometric structures.

In natural biological organisms, the processes of morphogenesis adapt structure to environment remarkably well on both an individual and evolutionary time scale. Great strides have been made in understanding the building blocks of biological adaptivity and the ties between

Work sponsored by DARPA DSO under contract W91CRB-11-C-0052; the views and conclusions contained in this document are those of the authors and not DARPA or the U.S. Government.

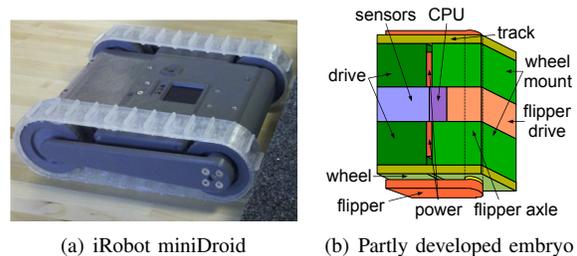


Fig. 1. We are investigating morphogenetic models as a representation for adaptable structures using the example of a “miniDroid” robot (a), a new member of the iRobot PackBot family shown in Figure 2. A preliminary test of our models is the creation of a program for early stages of development from an undifferentiated egg (b).

morphogenesis and evolution [6], [7], and there have been a number of systems that apply ideas from morphogenesis to particular aspects of design (see Section I-A). No clear framework has yet been developed, however, for exploiting morphogenetic principles in the creation of general engineered systems.

In this paper, we present preliminary work toward such a framework, using the example of design variations of a small ground robot, the iRobot miniDroid (Figure 1). More precisely, we consider the following problem: given a *design* known to function well in a particular range of *environments* and a set of *desired changes* to the design and/or its execution environments, compute either a new design satisfying the desired changes or a reason why the desired changes are infeasible.

We analyze how a morphogenetic model might serve as a reference architecture for the design of an engineered system, implicitly encoding constraints that allow many parameters of the design to be derived from a few key decisions. We suggest that this may allow for more flexible and efficient designs, where much of the process of adaptation can be carried out autonomously. Our morphogenetic model addresses shape, rather than control or behavior.

We then instantiate our morphogenetic model framework with a candidate basis set of five geometric operations, inspired by common processes in the embryogeny of animals, that can be composed together to form developmental programs. Such programs might be integrated together with functional blueprints [8] to facilitate design adaptation. Finally, we use the miniDroid example for preliminary validation of this approach, creating a program for development of a robot body plan, and discuss progress toward a software implementation.

A. Related Work

Automated design of electrical and/or mechanical systems has long been a topic of interest. While there has been much work in the area (e.g., [9], [10], [11]), it has faced significant obstacles from the complexity of searching the space of possible designs. A primary aim of our work is a principled reduction of this design space, such that the problem can be simplified.

This aim matches the goals of “morphogenetic engineering” as proposed by Doursat [12] and for robotics in [13] and [2]. Initial work towards a framework for morphogenetic engineering has been laid out for evolvable pattern formation in [14], and for embryogeny evolved through genetic programming in [15]. Meng et al.’s morphogenetic approach [16] to modular robotics combines a rule-based controller to generate desired patterns with a simulated genetic regulatory network to coordinate the configuration of robot modules. A more formal mathematical model can be found in [17], though the representational consequences are not explored.

A number of specialized languages for shape formation have been inspired by study of morphogenesis. For example, chemical gradients are the basis of Nagpal’s Origami Shape Language [18], which produces deformable geometric patterns, and Coore’s Growing Point Language [19], which forms topological patterns using a model of plant tropisms. Likewise, Kondacs’ model of pattern formation [20] creates regenerable patterns using a model of cell reproduction and apoptosis. To date, however, such languages have been specialized and hard to apply to the design of realistic engineered systems.

Finally, functional blueprints [21] are an engineering approach in which a system is specified according to its desired performance, with programs to incrementally adjust structure to improve performance when the goals are not met. The work presented in this paper is complementary, providing guidance on how functional blueprints could be best represented and applied.

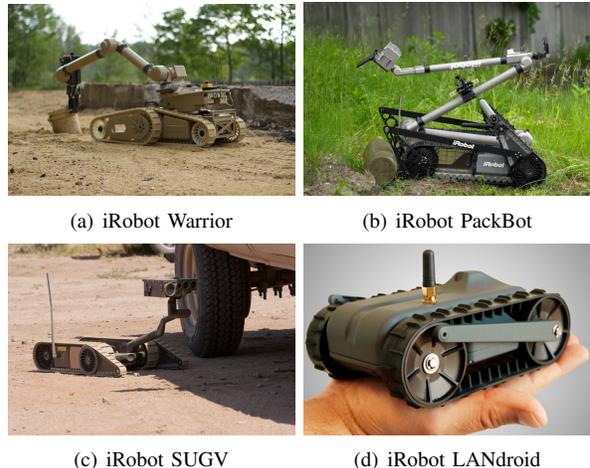


Fig. 2. Families of engineered systems often exhibit “phylogenetic” relationships similar to those of natural organisms. For example, these four iRobot products all share a base body plan, including symmetric two-wheel treads, flippers coaxial with one wheel, and a top-mounted sensor/manipulator package.

II. FROM NATURAL TO ENGINEERED MORPHOGENESIS

In order to base an engineering framework on natural morphogenesis, we must determine how methods drawn from the biological world are likely to provide a computational advantage over traditional engineering methods. Merely mimicking nature is not enough: after all, we would consider it foolish to add a heart or gender to a robot simply because animals have them. What then, is the advantage of morphogenesis?

To answer this question, we will consider variation within families of engineered designs, hoping to find systematic relations similar to that of natural systems [6]. In particular, we consider the family of products that the iRobot corporation has derived from their original PackBot system (Figure 2). Although these robots span a wide range of sizes and applications, all of them share a base body plan, including paired treads each driven by two wheels, flippers coaxial with one wheel, and a top-mounted sensor/manipulator package. We shall find that morphogenesis can act as a reference architecture, encoding relationships between design decisions and facilitating some types of variation at a cost of making others more difficult.

A. Morphogenesis Encodes Design Parameter Relations

In any system, some parameters strongly influence overall system design, while others can be derived by an expert human designer from the interaction of the key parameters. For example, the miniDroid, like all robots

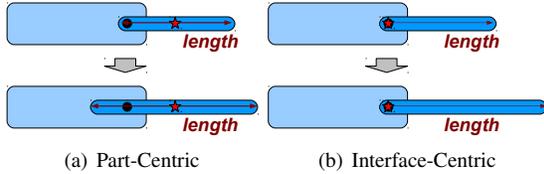


Fig. 3. Choice of coordinates for specifying a miniDroid flipper determines what happens when flipper length is increased. This is an example of how development order and coordinates can represent relationships that are intuitive to human designers.

in the PackBot family, uses its flippers to climb over obstacles. If the flippers are not long enough, the robot might be unable to climb high enough, so flipper length is important for this task. However, parameters that are indirectly related to functionality, such as the distance of the flipper from the geometric center of the robot’s body, can be derived from decisions about key parameters.

Consider a simple modification to the miniDroid shown in Figure 1(a). A designer might increase the miniDroid flipper length by 5mm, to allow the miniDroid to climb over slightly taller obstacles. When the length is increased, however, should the flipper extend symmetrically around its geometric center (Figure 3(a)), away from its attachment point (Figure 3(b)), or in any of the many other possible combinations? The answer is intuitive to a human designer, but a self-adapting design must have some way of representing this relationship.

This is potentially a severe problem: engineered designs generally have extremely high numbers of parameters: even a simple beam is described by at least nine parameters, including side lengths, position, and orientation. Only a few of these parameters are key, while most are instead constrained by their relationship to key parameters. Natural systems, however, must solve this problem in the developmental programs that create their forms, since evolution acts through relatively independent incremental changes to very small numbers of parameters at a time.

In animals, the developmental program effectively specifies a hierarchy of importance for spatial relationships between elements of the structure. For example, in mammals it is much easier for eyes to shift their location on the head than for an eye to shift to a different part of the animal. Likewise, the coordinate systems established during development establish an anisotropy on changes—when an arm is lengthened, it extends further out from the body, rather than attempting to invade into the body, and growth is distributed across the upper arm, forearm, and hand.

We thus look to morphogenesis as a means of encoding relationships between design parameters, defining a

developmental program as a partially-ordered sequence of operations over a spatial computer that convert a simple initial “egg” into a “mature” structure. If not terminated, such a program may also provide adaptation and repair for mature structures.

This developmental program also implicitly specifies an incremental process by which the elements of the design are related to one another, effectively creating a reference architecture—a document that aids in development of subsequent system designs by capturing design decisions and designer rationale. The reference architecture might then be leveraged by functional blueprints to simplify the creation of variant designs, while retaining the operational viability of each variant. Morphogenesis thus becomes a guide for how to maintain integration of a design as its components are being changed, whether by a human designer or an evolutionary algorithm.

The ordering of developmental operations establishes a hierarchy among the spatial relationships between elements of the system. Each operation creates one or more relations, and when the developmental program is modified, the earlier the modification, the more likely it is to have severe repercussions for the design. Conversely, later developmental stages are prime targets for variation. A developmental program thus dictates the difficulty of modifying different portions of the design. For example, if we specify the miniDroid flipper by growing it from a “limb bud” on the body, this creates interface-centric coordinates (Figure 3(b)) that will make it easy to extend the flipper away from its attachment point, but hard to extend it symmetrically around its geometric center. The process of developing a morphogenetic model is thus a matter of choosing which types of variation are expected to be desirable, and which types are to be discouraged.

B. Design Decisions for Applying Morphogenesis

In order to use this framework in engineering, there are two other required design decisions: how development is allocated across an organism’s life cycle, and how to select between mature and embryonic adaptation.

Most natural organisms undergo several qualitatively different stages of development. Engineered systems are likely to do the same, with some aspects of the design becoming well-defined while others are still highly plastic. For example, at present we are conceiving of robot development in three rough epochs. In the first epoch, the body plan is laid out, identifying all major components and their associated coordinate systems. In the second epoch, the design scales up such that components have the desired scale and refined to full detail, with functional blueprints beginning to act on the design. Finally, the design is executed solely under control of functional

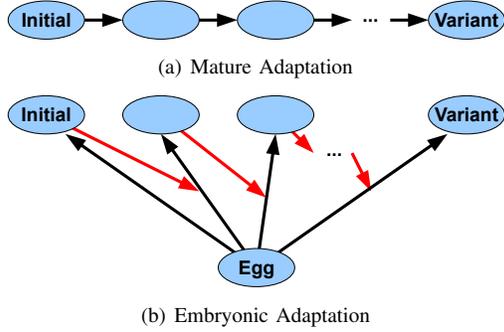


Fig. 4. A structure can adapt either directly in response to environmental feedback, similar to body changes within a mature organism (a), or by modifying the development of a real or virtual offspring (b).

blueprints in the full environment. These epochs do not have any prescriptive force but rather describe our thoughts on how to organize the developmental program for the miniDroid. However, we present them to show how the biological metaphor has continued to guide the design of our developmental program, and suggest that this may indicate a good area for further investigation.

Likewise, adaptation may take place either on an individual scale, in the continued development of a mature structure, or on an evolutionary scale, with new instances of a structure being created (Figure 4). The former has the advantage of being more incremental and less disruptive to a running engineered system, while the latter offers the potential of more radical adaptive change. Again, either model may be appropriate, depending on the circumstances of the engineered system.

III. DEVELOPMENTAL PROGRAMS FOR SPATIAL STRUCTURES

We now propose a representation of developmental programs: a candidate basis set of geometric operations, and an approach to handling fixed elements of a structure. The plausibility of this approach is demonstrated with an example developmental program for developing the body plan of a miniDroid from a square egg.

A. Geometric Operations

In natural systems, a vast array of different mechanisms are used to develop the form of an organism. From this panoply, we extract five simple abstract geometric operations for measuring, segmenting, and modifying manifolds. This choice reflects a conservative approach to developmental programs: each operation is based on an important and well-studied natural mechanism, and the set is likely to form a basis over all spatial structures, such that any structure can be expressed with an appropriate combination of operations. This is not necessarily

the “best” set of operations, but for preliminary work, we minimize the complexity of the model so that it will be easier to analyze its capabilities and implications. These operations work on particular tissues for which preconditions are met, and do not affect surrounding tissues.

Figure 5 illustrates our candidate set of operations:

- **coordinatize(0-region, 1-region)**: This operation computes a local coordinate system between two regions, ranging from 0 in one region linearly to 1 in the other region. It is based on the use of chemical gradients to polarize tissues, and most particularly on those where the signal is the ratio of two chemicals.
- **latch(region, type)**: The latch operation differentiates a region into a type. This operation is analogous to determination of cell fate in a biological system, such as the determination of limb fields.
- **scale(region, scale-factor)**: The scale operation takes a region and increases or decreases its scale proportionately along one or more axes. This is inspired by directed proliferation, such as is used in the construction of limb buds.
- **connect(source-region, destination-region)**: In this operation, a region extends from a source region along a near-shortest path to a destination region. This operation is based on cell migration, particularly the migration of tissue sheets, and on the extension of cell processes such as axons. We have tentatively specified that only the closest point in the source will connect, and that the width of the connection will be proportional to the size of the smaller region.
- **speckle(region, expected-separation)**: The speckle operation selects a set of small regions filling the space, each region determined randomly and separated by an expected distance from all others nearby. This is inspired by symmetry-breaking operations, such as those used to create hair follicles.

When composed using arithmetic, branching, and function definition, we believe that these operations may be sufficient to express most arbitrarily complex adaptable structures. Many other biological mechanisms can also be expressed as composites of these operations: for example, recruitment can be expressed as a coordinatize followed by a latch.

B. Fixed Structural Elements

While many elements of a structure can be treated as voxels that will assume arbitrary roles, some elements may be fixed in particular configurations that the developmental program must take into account. For example,

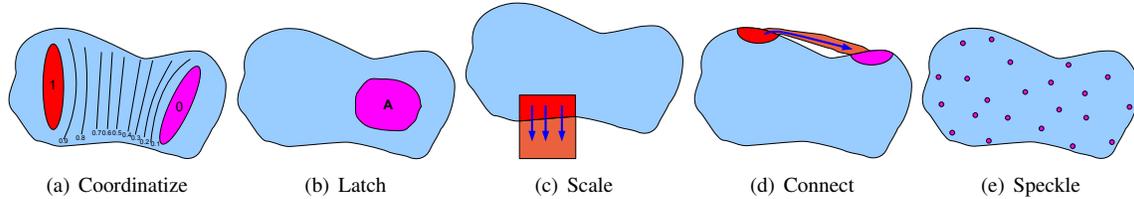


Fig. 5. We hypothesize that this set of five geometric operations may be a sufficient basis set for developing any design.

in designing a robot, we can assume that homogeneous materials like plastic or metal can be obtained in bulk and formed into arbitrary structures (e.g., via a 3D printer), but high-precision components like servo motors, bearings, and microcontroller boards are “packaged components” that come from suppliers in a limited number of models. Another example, in modular robotics, would be the difference between standard robots and specialized robots with larger dimensions.

This problem may be approached by arranging collections of related fixed elements in a space of their significant dimensions. For example, a collection of servo motors might be arranged by size, torque, and mass. A developmental program can then navigate through design space continuously in abstract, but always instantiate using the closest component. If there are enough elements in each collection, then it is unlikely that a design will become stuck in the gap between elements.

C. Example: Development of the miniDroid Body Plan

To validate the plausibility of our candidate operators, we have created a draft developmental sequence for the body plan of a miniDroid, which is likely to also apply well to other members of the PackBot family. The sequence begins with a square egg (to better match current engineered systems and manufacturing processes), and proceeds using the simplifying assumption that there are no mechanical fasteners. Our developmental sequence proceeds in eight stages, shown in Figure 6:

- Stage 1: basal coordinates are created, along anteroposterior, dorsoventral, and mediocentral axes. **Operations: coordinatize**
- Stage 2: basal coordinates are used to partition the robot into coarse body plan, with the exterior latching into “skin.” **Operations: latch**
- Stage 3: the electronics region differentiates along the anteroposterior axis into sensor and CPU regions. The boundary between limb buds recruits nearby material to form a gap between anterior and posterior limbs. **Operations: coordinatize, latch**
- Stage 4: limb buds scale themselves out of the body, then establish a proximodistal axis and lo-

cal centromedial axis. The power region recruits inter-limb gap material. **Operations: coordinatize, latch, scale**

- Stage 5: distal section of limb buds differentiates into wheels. Proximal section of front limb buds differentiates into drive. For rear limb buds, the flipper axle differentiates using the centromedial axis, and the proximal section of the limb bud differentiates into a passive mounting point. The wheels will round themselves later, as the body scales up. **Operations: latch**
- Stage 6: flipper axles scale in and out, meeting in the center. Wheel edges connect tracks between wheels. **Operations: scale, connect, latch**
- Stage 7: flipper axles scale outwards to form the flipper drive, pressing away nearby sections of CPU and power. Distal sections of axle differentiate into flipper buds and form a new anteroposterior axis. **Operations: scale, coordinatize, latch**
- Stage 8: flippers scale from buds. **Operations: scale**

At Stage 8, the complete body plan has been formed. Each of the sections now refines its details, while scaling up to reach a mature size. Meanwhile, power and signal wires connect the CPU, batteries, and motors using the connect operation, a process similar to innervation in vertebrates. If the fixed structural elements are handled using an approach like that described above, then an appropriate scaling of this body plan will produce a design similar to that of the miniDroid, and is expected to be simple to integrate with functional blueprints in order to enable largely autonomous adaptation.

IV. IMPLEMENTATION PROGRESS

We have begun a proof-of-concept Java implementation of our framework. Thus far, we have implemented a simple embryo model and two developmental operations, as described below, and used these to develop a robot embryo through stages 1-3 described above, providing preliminary evidence that our framework is feasible.

The developing embryo is represented as a collection of tissues, analogous to biological tissues (ensembles of cells of the same origin that carry out the same

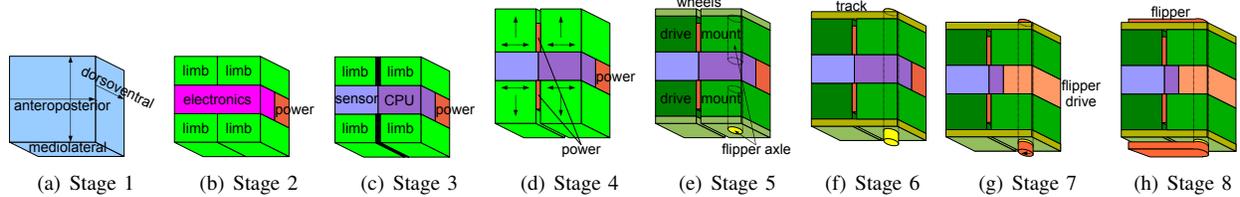


Fig. 6. A developmental program produces a body plan for a miniDroid (or other member of the PackBot family).

function). The tissues come in three types: stem cell (un-/partially-differentiated), voxelized (differentiated to a particular material, such as foam or aluminum) and packaged component (available in off-the-shelf varieties of various sizes and specifications, as with motors). Each tissue object contains a body part label (e.g., “limb bud”), a type, and a “geometry” that specifies shape, coordinates, and means of converting between its own coordinate system and others.

The morphogenetic process is managed by a developmental program, containing the embryo and a list of developmental rules. The embryo starts as a square egg of undifferentiated tissue. Each rule contains an operation, an execution function, and tests for any preconditions. With each timestep, all rules whose preconditions are met are executed as a batch. Thus far, we have implemented the development of the miniDroid embryo through three of the eight stages described earlier.

V. CONTRIBUTIONS AND FUTURE WORK

We have presented a preliminary morphogenetic engineering framework, in which we exploit the implicit spatial relations of developmental programs to provide adaptability in the structure of a spatial computer. Driven by the example of varying the design of an iRobot miniDroid, we have instantiated this framework with a candidate basis set of five geometric operations, and partially implemented it in software.

Immediate next steps are focused on completing the software implementation and validating the framework by integrating it with functional blueprints to produce functional variations of robotic designs. More generally, this framework may provide a means, across the full range of spatial computers, of making shape formation more flexible, and design variation simpler.

REFERENCES

- [1] J. Werfel and R. Nagpal, “Collective construction of environmentally-adaptive structures,” in *Int’l Conf. on Intelligent Robots and Sys.*, 2007.
- [2] R. O’Grady, A. L. Christensen, C. Pinciroli, and M. Dorigo, “Robots autonomously self-assemble into dedicated morphologies to solve different tasks,” in *Int’l Conf. on Autonomous Agents and Multiagent Systems (AAMAS)*, 2010.
- [3] J. McLurkin, “Stupid robot tricks: A behavior-based distributed algorithm library for programming swarms of robots,” Master’s thesis, MIT, 2004.
- [4] C.-H. Yu and R. Nagpal, “Self-adapting modular robotics: A generalized distributed consensus framework,” in *International Conference on Robotics and Automation (ICRA)*, 2009.
- [5] R. Langer and J.P. Vacanti, “Tissue engineering,” *Science*, vol. 260, no. 5110, pp. 920–926, May 1993.
- [6] S. B. Carroll, *Endless Forms Most Beautiful*. W. W. Norton & Company, 2005.
- [7] M. W. Kirschner and J. C. Norton, *The Plausibility of Life: Resolving Darwin’s Dilemma*. Yale University Press, 2005.
- [8] J. Beal, “Functional blueprints: An approach to modularity in grown systems,” *Swarm Intelligence*, vol. 5, no. 3, 2011.
- [9] M. I. Campbell, J. Cagan, and K. Kotovsky, “A-design: An agent-based approach to conceptual design in a dynamic environment,” *Research in Engineering Design*, vol. 11, pp. 172–192, 1999.
- [10] J. d. K. Markus P. J. Fromherz, Daniel G. Bobrow, “Model-based computing for design and control of reconfigurable systems,” *AI Magazine*, vol. 24, no. 4, 2003.
- [11] S. P. Hoover and J. R. Rinderle, “A synthesis strategy for mechanical devices,” *Research in Engineering Design*, vol. 1, pp. 87–103, 1989.
- [12] R. Doursat, “Morphogenetic engineering weds bio self-organization to human-designed systems,” *PerAda Magazine*, 2011.
- [13] Y. Jin and Y. Meng, “Morphogenetic robotics: An emerging new field in developmental robotics,” *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, vol. 41, no. 2, pp. 145–160, 2011.
- [14] R. Doursat, “Organically grown architectures: Creating decentralized, autonomous systems by embryomorph engineering,” in *Organic Computing*, R. Wurtz, Ed. Springer-Verlag, 2008, pp. 167–200.
- [15] P. Bentley and S. Kumar, “Three ways to grow designs: A comparison of embryogenies for an evolutionary design problem,” in *Genetic and Evolutionary Computation Conference*, 1999.
- [16] Y. Meng, Y. Zhang, and Y. Jin, “A morphogenetic approach to self-reconfigurable modular robots using a hybrid hierarchical gene regulatory network,” in *International Conference on the Synthesis and Stimulation of Living Systems*, 2010.
- [17] B. MacLennan, “Models and mechanisms for artificial morphogenesis,” in *Natural Computing, Proceedings in Information and Communications Technology*, vol. 2, 2010.
- [18] R. Nagpal, “Programmable self-assembly: Constructing global shape using biologically-inspired local interactions and origami mathematics,” Ph.D. dissertation, MIT, 2001.
- [19] D. Coore, “Botanical computing: A developmental approach to generating inter connect topologies on an amorphous computer,” Ph.D. dissertation, MIT, 1999.
- [20] A. Kondacs, “Biologically-inspired self-assembly of 2d shapes, using global-to-local compilation,” in *International Joint Conference on Artificial Intelligence (IJCAI)*, 2003.
- [21] A. Adler, F. Yaman, J. Cleveland, and J. Beal, “Morphogenetically assisted design variation,” in *International Conference on Morphological Computation (in press)*, 2011.