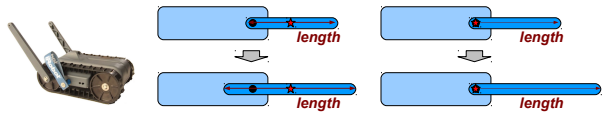
Morphogenesis as a Reference Architecture for Engineered Systems

Jacob Beal[1] Annan Mozeika[2] Jessica Lowell[1] Kyle Usbeck[1]
[1]BBN Technologies 10 Moulton Street Cambridge, MA, USA 02138
[2]iRobot Corporation 8 Crosby Drive Bedford, MA, USA 01730
{jakebeal,jlowell,kusbeck}@bbn.com, amozeika@irobot.com

Morphogenetic models for engineering are an attractive idea, as current engineered systems tend to be brittle in their design, while biological organisms adapt structure to environment remarkably well on both an individual and evolutionary time scale. The continuing synthesis of biological evolution and development (see [2, 3]) strengthens the case for attempting to extract new engineering principles from the mechanisms of morphogenesis. We prefer, however, not to engage in mere biomimicry, but to ask: what are the concrete *representational* advantages that are derived from a morphogenetic approach?

We hypothesize that a developmental program for an engineered design can improve its flexibility by reducing the redundant degrees of freedom in a design. 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. A developmental program 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—that is otherwise lost in the blueprints of a "mature" system. The reference architecture might then be leveraged by functional blueprints [1] 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.



(a) LANdroid (b) Part-Centric (c) Interface-Centric

Figure 1: A developmental program for a robot breaks symmetry, determining how changing an attribute (e.g. flipper length) interacts with other aspects of a design.

We are investigating this hypothesis in the context of designing a robot similar to the iRobot LANdroid (Figure 1(a)), which uses a pair of limb-like flippers to help it climb over obstacles. Consider, for example, an incremental increase in the length of the flipper, to help the robot climb over a larger obstacle. That length increase can be realized in many ways: the flipper can extend forward, backward, or any mixture of the two. In addition,/ the flipper axle's attachment point can remain fixed with respect to the robot body, with respect to its distance from either the front or back of the flipper, or any mixture of these. Even such a simple change has many options and no *a priori* means of distinguishing between them.

Specifying a developmental program implicitly specifies a hierarchy of importance for spatial relationships between elements of the structure. These relationships in turn create a design-specific atlas of coordinates that dictate how variations interact. For example, when a hominid arm is lengthened, the developmental program for limb buds ensures that 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. Figure 1 illustrates this idea for the example of changing flipper length on a robot, the difference between "part-centric" coordinates typically used in engineering blueprints, and an "interface-centric" coordinates that might be specified in a developmental program.

Each choice in a developmental sequence implies that some modifications will be easier, and others will be harder. By choosing a particular sequence, we are making commitments about its position in a "phylogenetic tree" of possible variants. The developmental sequence thus effectively encodes the relative *priority* of design commitments. Decisions that are encoded earlier in the developmental process are more difficult to change in variant designs because once they are encoded, they may affect part relationships in later stages. Conversely, late-developing structures or relationships will tend to be more independent of one another and therefore easier to modify. For example, a LANdroid-like robot design might begin with a body plan based around four limbs, which later differentiate into wheels and connect, or around two tracked drives that end up differentiated to contain two wheels each. By encoding the "limb vs. track" decision at such an early stage, we make it a fundamental choice about the nature of the design, and difficult to modify in subsequent variations. Thus, these two choices each imply a different family of easily accessible variants (Figure 2).

We have created a draft developmental sequence for the body plan of a LANdroid-like robot. The sequence begins with a square egg (to better match current engineered systems and manufacturing processes), and proceeds using two simplifying assumptions: there are no mechanical fasteners,

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.
3rd Morphogenetic Engineering Workshop 2011 Paris, France

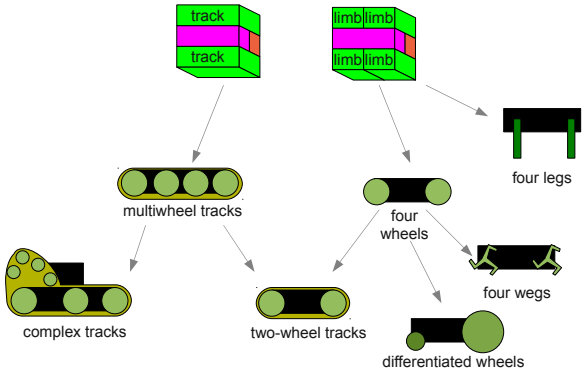


Figure 2: Alternate developmental paths for LANdroid-like robot exist within different families of easily accessible variants.

and elements can be formed of arbitrary substance.

Our developmental sequence constructs shape with the aid of four manifold operations based on well-understood patterns in biological morphogenesis: coordinatize (chemical gradient), latch (selection of cell fate), scale (directed proliferation), and connect (cell migration). We apply these operators to produce the sequence of developmental stages shown in Figure 3.

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 a coarse body plan, with the exterior latching into "skin" and the borders between regions latching to become"frame." *Operations: latch*

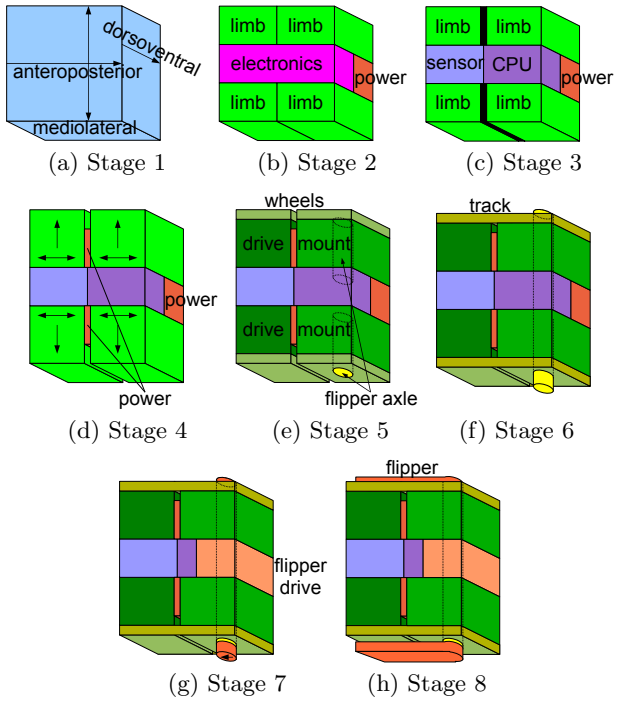
Stage 3: the electronics region differentiates along the anteroposterior axis into sensor and CPU regions. The frame between limb buds recruits nearby material to form a gap between anterior and posterior limbs. *Operations: coordinatize, latch*

Stage 4: limb buds scale themselves out of the body, then establish a proximodistal axis and local centromedial axis. The power region recruits inter-limb gap material. *Operations: coordinatize, latch*

Stage 5: distal section of limb buds differentiates into wheels. The 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 to meet in center and form new buds at the distal end. Wheel edges use chemotaxis to connect tracks from the front wheel to the back wheel. *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 the axle differentiate into flipper buds and form a new anteroposterior axis. *Operations: scale, coordinatize, latch*



(a) Stage 1 (b) Stage 2 (c) Stage 3

(d) Stage 4 (e) Stage 5 (f) Stage 6

(g) Stage 7 (h) Stage 8

Figure 3: Developing body plan for a LANdroid-like robot with flippers.

Stage 8: flipper buds form flippers. *Operations: scale*

After stage 8, the base robot body plan has been formed. Each of the sections refines its details, scaling to a mature size, as dictated by functional blueprints. Meanwhile, power and signal wires connect the CPU, batteries, and motors using the connect operator, a process similar to innervation in vertebrates. We are currently working to integrate this developmental sequence with functional blueprints that we have developed for the robot, such that the coordinate systems created during development can be used to guide variation as described above.

In summary, we hypothesize that developmental programs can act as reference architectures by concisely expressing the designer's decisions while maintaining adaptability toward subsequent designs. The encoding of a developmental model may then organize the relative difficulty of modifying each design decision—a mechanism for scoping designer rationale to particular design decisions. The presented robot design examples represent steps toward testing these hypotheses in the context of complete engineered systems.

References

[1] J. Beal. Functional blueprints: An approach to modularity in grown systems. In *International Conference on Swarm Intelligence*, 2010.

[2] S. B. Carroll. *Endless Forms Most Beautiful*. W. W. Norton & Company, 2005.

[3] M. W. Kirschner and J. C. Norton. *The Plausibility of Life: Resolving Darwin's Dilemma*. Yale University Press, 2005.