

Mixed Geometric-Topological Representation for Electromechanical Design *

Jacob Beal
Raytheon BBN Technologies
Cambridge, MA, USA 02138
jakebeal@bbn.com

Aaron Adler
Raytheon BBN Technologies
Cambridge, MA, USA 02138
aadler@bbn.com

Hala Mostafa
Raytheon BBN Technologies
Cambridge, MA, USA 02138
hmostafa@bbn.com

ABSTRACT

Avoiding unintended representational commitments is a key challenge in generative design. We have developed a mixed geometric-topological representation based on CW-complexes, which represents structure and geometric constraints such that commitments regarding position and layout are late-binding and resolve only during the evaluation of a design instance. Complicated designs can be elaborated into a full representation using a small number of biologically-inspired developmental operators. We illustrate the new representation with a number of examples of electromechanical design.

Categories and Subject Descriptors

J.6 [Computer-Aided Engineering]

Keywords

Morphogenetic Engineering, Generative Design, Implicit Embryogeny, Generative Representation

1. MOTIVATION

A key challenge in generative design is to find representations that allow smooth navigation of the design space. Natural biological organisms, which are remarkably adaptable both as individuals and in evolving populations over time, derive much of their adaptability from the process of morphogenesis [4, 8], in which the basic topological structures and relations laid out in these early stages of development then persist throughout development, even as the various body parts and organs assume their full-grown dimensions.

Prior development-inspired representations for electromechanical design (e.g., [7, 3, 6, 5, 2]) represent topological relations and constraints implicitly, through the geometric simulation of development. This is often computationally expensive, and in many cases results in unintended dependencies or representational commitments imposed by the progress of a design through intermediate forms on its way to a “mature” state where the design can be evaluated.

We propose instead to base a representation on topology and attach geometric parameters and constraints to this model only as they become relevant, creating a “late-binding” geometry that preserves flexibility and symmetry

*Sponsored by DARPA under contract W91CRB-11-C-0052; views and conclusions contained in this document are those of the authors and not DARPA or the U.S. Gov’t.

except where explicitly broken and allows constraints and dependent parameters to be specified implicitly.

2. HYBRID REPRESENTATION

We have developed a hybrid topological-geometric representation in four layers of objects and directed binary relations. The foundation layer is a construct of topological cells, specifying the collection of geometric components making up a design and their attachment and containment relations. We based this on a topological object known as a CW-complex [9], a topological cousin to familiar geometric engineering constructs such as the polygonal mesh and thus a natural fit for representing electromechanical designs.

To this topological base, we attach numerical parameters and measurements. In some cases, these parameters represent aspects of geometry, and thereby add constraint to the topology, while in other cases they associate other types of specification information (e.g., material properties). Each parameter’s units are specified, though its value is not set at this layer of the representation.

The third layer is a collection of relations that constrain the values of parameters, either specifying physical relationships between parameters or connecting parameters to fitness evaluations. These relations effectively reduce the number of degrees of freedom in the system of parameters, though exactly how much may not always be clear, since some relations may be partially or entirely redundant.

Finally, a collection of values is assigned to parameters, some of which may be directly specified, while others may be derived from the constraints between parameters. Propagating these values through the relations on parameters can then fill in unspecified parameters and resolve conflicts between parameters (e.g., via functional blueprints [1]), completing the specification of a design.

Taken together, these four layers form a representation in which it is possible to specify exactly what is known or constrained within a design, with no requirement to fix any value that has no reason to be fixed. This avoids the premature commitment or over-constraint that often appears in geometry-based representations.

3. DEVELOPMENTAL PROGRAMS

Designs can be specified succinctly in our representation by a developmental program that applies successive transformations to a trivial “seed” topology to produce the final representation. The “genotype” of a design is then the sequence of developmental operators used to develop a design from an initial point. For an initial set of developmental

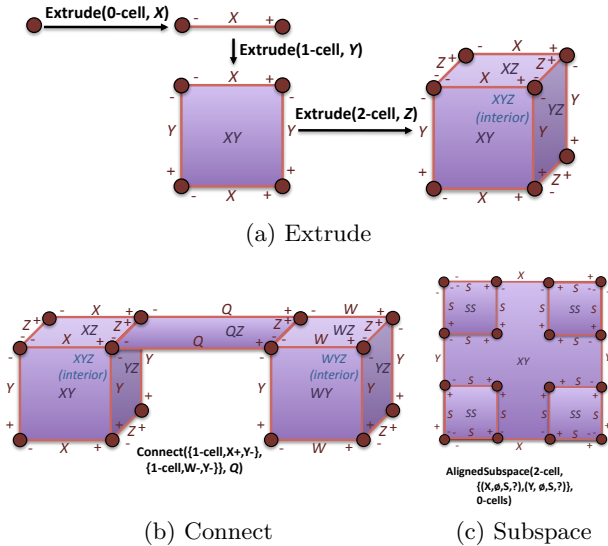


Figure 1: Example constructions of (a) a 3D rectangle from three Extrudes, (b) linking X+,Y- and W-Y- 1-cells with Connect, and (c) four symmetric regions from a corner-anchored AlignedSubspace.

operators, we consider a set of five operators based on the set in [2]. These operators and their parameters are:

- **Extrude(cells,coordinate):** For every k -cell (a k -dimensional space homeomorphic to the unit ball) in the set **cells**, extend that cell to create a new orthogonal axis measured by **coordinate**.
- **Connect(-cells,+cells,coordinate):** For every k -cell in the set **-cells**, create a $(k + 1)$ -cell measured by **coordinate** that connects to it from the k -cell in **+cells** with the most similar values of other coordinates.
- **AlignedSubspace(cells,coordinate-map,anchors):** Create one or more subspaces within each element of **cells**, with the relative position and orientation of the subspaces set by the other parameters.
- **RelateParameters(relation,parameters):** Link the set of **parameters** by a new instance of **relation**.
- **AssignValue(parameter,value):** Create **value** and link it to **parameter**.

With these operators, complex topological constructs, parameter sets, and relations can be generated with a few simple operations, and using predicate matching to determine application can allow adaptation to upstream changes in design. Developmental programs thus compress a design into a compact form containing no redundancies and relatively little interdependence between statements.

A design is then fully instantiated by using the constraints and values assigned to geometric parameters to determine a set of permissible positions of topological elements. While this is not guaranteed to be possible, if positions are seeded from the last viable layout and resolved using spring forces, then resolution of positions is likely to rapidly find an acceptable solution across a wide range of common cases.

4. SIMPLE DESIGN EXAMPLES

We have implemented a version of the mixed topological-geometric representation and the developmental operators using Lisp. Using this implementation, we have constructed representations for several simple examples of designs:

- One or two solar panels on a roof.
- A Latin cross (three arms symmetric, bottom long)
- A table with four symmetric legs
- A chair with four symmetric legs and a raised back

For each example, we illustrate the power of our representation by comparing the number of topological operators required to generate the design with geometric representations in terms of points, edges, or convex space partition:

Example	Operators	Points	Edges	Convex Spaces
One Solar Panel	3	8	8	5
Two Solar Panels	4	12	12	*
Latin Cross	6	24	44	5
Table	5	36	60	11
Chair	7	40	68	*

* impossible without additional assumptions

Our representation is much more terse than geometric points or edges. Convex partitions are more succinct, but even simple designs often cannot be represented without adding unwarranted assumptions about relative position (e.g., of two solar panels) or size (e.g., of chair back vs. legs).

5. FUTURE DIRECTIONS

To date this new representation has only been applied to relatively simple designs; the obvious next steps are to apply it to more complex designs and to formally evaluate its benefit in design space smoothness relative to geometric representations, particularly regarding exploration of structural change by modification of the developmental program.

Other future directions include using this representation in design adaptation engines such as evolutionary algorithms, functional blueprints [1], logical reasoning, or numerical optimization. Further possibilities for generalization and extension of the representation include representation of polygonal and curved structures, additional bio-inspired developmental operators, context-driven application of development rules, and non-physical structures such as control rules.

6. REFERENCES

- [1] J. Beal. Functional blueprints: An approach to modularity in grown systems. *Swarm Intel.*, 5(3), 2011.
- [2] J. Beal, H. Mostafa, A. Mozeika, B. Axelrod, A. Adler, G. Markiewicz, and K. Usbeck. A manifold operator representation for adaptive design. In *GECCO 2012*, July 2012.
- [3] J. C. Bongard and R. Pfeifer. Repeated structure and dissociation of genotypic and phenotypic complexity in artificial ontogeny. In *GECCO*, pages 829–836, 2001.
- [4] S. B. Carroll. *Endless Forms Most Beautiful*. W. W. Norton & Company, 2005.
- [5] R. Doursat. Organically grown architectures: Creating decentralized, autonomous systems by embryomorph engineering. In R. Wurtz, editor, *Organic Computing*, pages 167–200. Springer-Verlag, 2008.
- [6] P. Eggenberger. Evolving morphologies of simulated 3d organisms based on differential gene expression. In *ECAL*, pages 205–213. MIT Press, 1997.
- [7] G. S. Hornby and J. B. Pollack. Creating high-level components with a generative representation for body-brain evolution. *Artificial Life*, 8:223–246, 2002.
- [8] M. W. Kirschner and J. C. Norton. *The Plausibility of Life: Resolving Darwin's Dilemma*. Yale U. Press, 2005.
- [9] J. H. C. Whitehead. Combinatorial homotopy I. *Bull. Amer. Math. Soc.*, 55:213–245, 1949.