Morphogenesis as a Reference Architecture for Engineered Systems

Jacob Beal, Annan Mozeika, Jessica Lowell, Kyle Usbeck

August 12, 2011

3rd Morphogenetic Engineering Workshop @ ECAL ‘11

Work sponsored by DARPA; the views and conclusions contained in this document are those of the authors and not DARPA or the U.S. Government.
How to grow a robot from an egg…

… and why it matters with current manufacturing
Morphogenetically Assisted Design Variation

- An interactive design tool providing non-experts the ability to vary robot design
  - User modifies evaluation parameters e.g., conditions of an obstacle course, and robot design is automatically varied to cope with new situation

MADV varies robot design enabling it to climb larger steps

User indicates robot should be able to climb larger steps
Design Challenges

- What needs to change for new conditions?
- How does a change impact other systems?
Problem: Propagating Changes

A small change ...

Extending Flippers
Problem: Propagating Changes

A small change ... has many consequences
Morphogenesis enables natural variation

Felidae

Felidae

Felinae

Felidae

Pantherinae

Panthera

Felis

Cheetah

Snow Leopard

Panthera

Pantherinae

Felidae

Felinae

Felis

Domestic Cat

Pallas Cat

Tiger

Lion
A phylogeny of engineered systems?

PackBot → LANdroid → miniDroid → SUGV → Warrior
Our Approach: Functional Blueprints

- Functional blueprints specify design as functional goals and a means to adjust the structure when the goals are not met.

- Functional blueprints capture dependencies that are hard to represent with traditional blueprints.

- Stress is used as the coordinating signal driving structural changes.
Functional Blueprint Definition

1. Functional behavior that degrades gracefully
2. Metric for degree and direction of stress
3. Incremental adjustment program for stress relief
4. Initial viable system
Functional Blueprints: Stress Functions

- Idea: keep the design always working, navigate through viable space
Functional Blueprints: Stress Functions

- Idea: keep the design always working, navigate through viable space

- Stress functions define viable and non-viable space
Functional Blueprints: Previous Results

- Functional blueprint model of vascularization
  - Stress: oxygen, elastic stress
  - Adjustment: leaking, vessel grow/shrink
- Red cells are healthy, blue cells are oxygen-deficient
- Can model vasculatization and density co-regulation
Moving to robots...

- How do we apply preliminary work to electromechanical systems?
  - How to evaluate function?
  - How can the functional blueprints be composed?
  - How to safely integrate user input?
  - What types of stress and update functions make sense?
  - How to transform the design into a variant?
Initial Target: Step Climbing

• Required extending initial functional blueprint model
  – Added indirect stress generation: evaluation of design includes system properties and derived properties, e.g., time to accomplish a task
  – Define stress as a function of evaluator outputs
• Robot design evaluated using Open Dynamics Engine (ODE) simulation
• Robot Model is based on miniDroid design
• Initial indirect stress functional blueprint: step climbing
Stair Climbing Metrics

• Currently using body angle at “critical point” – flipper perpendicular to ground
• Good metric choice is critical
  – e.g., climbing time not good because failure and stress not well separated.

Stair Climbing Metrics (5 unit long robot)

![Graph showing stair climbing metrics](image-url)
Where are Functional Blueprints used?

- To represent robot capabilities
  - e.g., FBs for climbing a step and flipping over
- To safely integrate user requested changes
  - Jumping to the user desired value may put the design in non-viable space
  - Generic, parametric perturbation FBs created on the fly to incrementally incorporate user requested changes
- To represent constraints enforced by component libraries
  - e.g., the size and torque of a servo required by the design must match a servo in the component library

Copyright 2011 Raytheon BBN Technologies
# Functional Blueprints for MiniDroid

<table>
<thead>
<tr>
<th>FB</th>
<th>Stress Function</th>
<th>Update Functions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Max Torque</td>
<td><img src="image" alt="Graph" /></td>
<td>Decrease torque</td>
</tr>
<tr>
<td>Flipping</td>
<td></td>
<td>Increase torque</td>
</tr>
<tr>
<td>Body Proportion</td>
<td></td>
<td>Decrement flipper length</td>
</tr>
<tr>
<td>Climb</td>
<td><img src="image" alt="Graph" /></td>
<td>Increase flipper and body length</td>
</tr>
<tr>
<td>Servo Viability</td>
<td><img src="image" alt="Graph" /></td>
<td>Decrease torque and increase motor mass</td>
</tr>
<tr>
<td>Servo Mass</td>
<td><img src="image" alt="Graph" /></td>
<td>Increase/decrease motor mass</td>
</tr>
<tr>
<td>Perturbation on X</td>
<td><img src="image" alt="Graph" /></td>
<td>Increase/decrease perturbed attribute X</td>
</tr>
</tbody>
</table>

19
MADV Prototype with Seven FBs

Robot Model

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>left arm length</td>
<td>0.2</td>
<td>meters</td>
</tr>
<tr>
<td>motor mass</td>
<td>23.81</td>
<td>grams</td>
</tr>
<tr>
<td>motor torque</td>
<td>3.5</td>
<td></td>
</tr>
<tr>
<td>body z</td>
<td>0.375</td>
<td>meters</td>
</tr>
<tr>
<td>left arm y</td>
<td>1.11</td>
<td>meters</td>
</tr>
</tbody>
</table>

Environment Model

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
<th>Units</th>
<th>Desired Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>step length x</td>
<td>10</td>
<td>meters</td>
<td></td>
</tr>
<tr>
<td>step length y</td>
<td>10</td>
<td>meters</td>
<td></td>
</tr>
<tr>
<td>step x</td>
<td>7</td>
<td>meters</td>
<td></td>
</tr>
<tr>
<td>step length z</td>
<td>3.3</td>
<td>meters</td>
<td>3.3 meters</td>
</tr>
<tr>
<td>step mass</td>
<td>10</td>
<td>grams</td>
<td></td>
</tr>
</tbody>
</table>

Functional Blueprint Stresses

Total System Stress: 0.5 --- Stress for Feature [name=MotorMass FB] - 0.0 ; Stress for Feature [name=Proportion FB] - 0.6 ; Stress for Feature [name=MotorViability FB] - 0.0 ; Stress for Feature [name=Flipping FB] - 0.0 ; Stress for Feature [name=Climb FB] - 0.0 ; Stress for Feature [name=MaxTorque FB] - 0.0 ; Total Stress - 0.5 ; FB step length z: 2.05 meters

Chose hs=125mg for component motor
MADV Prototype with Seven FBs

Initial simulation:

Simulation after changes:

*Higher step ➔ longer flipper, longer body, bigger motor*
How Hard is it to Change Flipper Length?
How Hard is it to Change Flipper Length?

Does it grow from the center?
How Hard is it to Change Flipper Length?

Does it grow from the center? the front?
How Hard is it to Change Flipper Length?

Does it grow from the center? the front? the back?
How Hard is it to Change Flipper Length?

Does it grow from the center? the front? the back?

Functional blueprints control the key attributes…
… but our designs have many others!
A Complex Transformation…

• Both flippers are driven by one servo…
A Complex Transformation…

• Both flippers are driven by one servo…
  … what if the robot widens to need two?
Solution: Developmental Program

A developmental program constrains geometric relationships between components

- Reduced dimensionality
- Greater design flexibility
Developmental Primitives

We begin with 5 manifold operations:

- Coordinatize
- Latch
- Scale
- Connect
- Speckle

Each based on a key animal development pattern.

*How far can we get with just these?*
Developmental Program for Body Plan

Final size: ~2mm
Next: prenatal growth
Body Plan Dictates Family of Variants

- The sequence of development for a body plan implies a prioritization of major design features, selecting a family of more accessible variants.
Other Alternative Body Plans

- Symmetric power or electronics splits a unified component
- Nested body adds ill-motivated complexity
- Of the asymmetries, ventral power also implies a preferred low center of gravity
Other Key Developmental Decisions

- Wheel attachments: packaged component with a “base limb” included within wheel
- Flippers: based on wheel axles ensures flipper/wheel structure integrity, but makes them hard to separate
Program Representation: Manifold Rules

- Parallel application, continuous manifold evolution, conflict resolution by actuator blending
- Benefits: implicit relations, easy to modify/insert
Program Representation: Manifold Rules

- Parallel application, continuous manifold evolution, conflict resolution by actuator blending
- Benefits: implicit relations, easy to modify/insert
Details: Packaged Components

**Approach:** interpolate across parts from a “Component Model Library”
Details: Wiring

- Chemotactic model:
Contributions & Next Steps

• Functional blueprint concept adapted for electromechanical design
• Development sets implicit structural relations
• Initial developmental models for miniDroid

Next: full simulation, integration with FBs
Goal: development of novel structures
Project Team:

**Raytheon**
BBN Technologies
Jacob Beal (PI)
Aaron Adler (co-PI)
Susan Katz (PM)
Brett Benyo
Jeff Cleveland
Jessica Lowell
Hala Mostafa
Kyle Usbeck
Fusun Yaman

Interns:
• Katie McGuire
• Taylor Campbell

**iRobot**
Annan Mozeika

Intern:
• Gretchen Markiewicz

Team website:
http://madv.bbn.com

Sponsored by:

Copyright 2011 Raytheon BBN Technologies