2013 IEEE/RSJ International Conference on
Intelligent Robots and Systems (IROS)
September 14-18, 2014. Chicago, IL, USA,

Cogeneration of Mechanical, Electrical, and Software Designs for Printable Robots from Structural Specifications

Ankur M. Mehta, Joseph DelPreto, Benjamin Shaya, and Daniela Rus
{mehtank, delpreto, bshaya, rus}@csail.mit.edu

Abstract— **Designing and fabricating new robotic systems is typically limited to experts, requiring engineering background, expensive tools, and considerable time. In contrast, to facilitate everyday users developing custom robots for personal use, this work presents a new system to easily create printable foldable robots from high-level structural specifications. A user merely needs to select electromechanical components from a library of basic building blocks and pre-designed mechanisms, then connect them to define custom robot assemblies. The system then generates complete mechanical drawings suitable for fabrication, instructions for the assembly of electronics, and software to control and drive the final robot. Several robots designed in this manner demonstrate the ability and versatility of this process.**

I. Introduction

While robots are widely used for research and commercial applications, they are not yet ubiquitous in everyday life for personalized tasks. Creating a new robotic system typically involves repeated design iterations using a variety of computer-aided design tools. Domain-specific expertise is generally required to create the requisite mechanical drawings for a structural body, electronic circuits to connect sensors and actuators, and software to manage system inputs and outputs. This entire process often needs to be rerun for each new robot desired, making the design and fabrication of new robots slow and introducing a knowledge barrier for casual users.

For the general public to obtain a device able to accomplish a specified function, computational tools are required that can create robots from high-level descriptions. The long-term objective is to develop a hardware compiler that can start with functional specifications of a desired system and automatically design and fabricate a robot to accomplish those. This paper takes a step towards that vision with a system to simultaneously generate the mechanical, electrical, and software components of a robot from its structural specifications, allowing non-experts to easily design electromechanical systems with custom specifications and then quickly and inexpensively fabricate the designed robot.

The presented approach modularizes mechanical, electrical, and software components to represent them in a database suitable for hierarchical composition. These parameterized components are characterized in terms of their geometric and physical properties; while experts can directly generate new low level building blocks, casual users can simply connect existing designs to make custom electromechanical devices. The system then automatically outputs a collection of files with which the user can manufacture the specified robot: fabrication drawings get sent to a desktop cutter to generate cut-and-fold origami-inspired 3D structures, wiring instructions guide the user to assemble sensors and actuators onto that structure using plug-and-play electronic modules, and libraries and application software get loaded onto a central microcontroller. The resulting robot can be wirelessly controlled from an auto-generated user interface (UI) on a smartphone.

This work extends the previous work reported in [1], [2], which focused only on mechanical structures, to create a system which directly translates structural specifications into a fully functional printable robot. In particular, this paper presents the following:

- a system which encapsulates mechanical, electrical, and software designs into parameterizable self-contained components with well-defined interfaces for composition with other such components,
- a process to hierarchically compose those elements into electromechanical mechanisms of arbitrary complexity,
- a library of base and derived components, and
- several robots designed, fabricated, and operated using the proposed system.

This paper begins in section II with a brief look at related research. A detailed breakdown of the desired goals of this work is presented in section III, followed by a description of the implementation in section IV. Several case studies in which this system was used to design and fabricate a variety of robots follow in section V. The paper concludes with an analysis of the benefits of this system and plans for future work in section VI.

II. Related Work

This paper builds on a body of work developing processes to rapidly fabricate robots. In particular, 2D processes have been used to create 3D structures [3], [4] and robots [5]–[7] in various mediums across a range of size scales. These fabrication methods have been employed for rapid prototyping (e.g. in [8]), being able to produce devices in a timeframe of minutes. However, creating robots with these processes typically requires careful hand design by experienced engineers.

There have been attempts to automate the decomposition of 3D shapes, notably in [9]–[11], to generate 2D fold patterns. These tools and algorithms, however, focus mostly on solid objects, employing various heuristics to generate polyhedra obeying certain rules. Compliant and kinematic structures are not addressed.

978-1-4799-6933-3/14/$31.00 ©2014 IEEE

The system presented in this paper extends earlier work on simplifying the design of origami inspired cut-and-fold robotics. Manually scripted elements were used to simplify design iteration in [1] to create folded micro air vehicles, while component-based mechanical design was demonstrated in [2].

III. System goals

Robot creation traditionally consists of a sequence of phases during which mechanical, electrical, and software subsystems are designed and then integrated. This process must be largely started anew for each distinct robot, requiring the designer to have a deep understanding of the interplay between the separate subsystems as they relate to the robot as a whole. Instead, the system presented in this work leverages a modular paradigm which allows electrical, mechanical, and software components to be coupled at the lowest level. As a user combines these larger electromechanical modules, subsystem designs are assembled behind the scenes to maintain an integrated design throughout the modular hierarchy. This high-level layout of modules can thus be compiled to generate fabrication specifications which directly produce the robot.

In order to realize this system, a library of modules must be created by expert users that encapsulates mechanical and electrical components. In [2], a modular method was proposed for specifying mechanisms by combining structural elements with compliant degrees of freedom. Analogous electrical blocks can now be defined using the same paradigm: expert designers create the building blocks with parameterized specifications and allowable interfaces for composition. Modularizing these systems gives the ability to group the mechanical and electrical blocks into higher level modules. Users can then combine basic blocks into complex electromechanical systems, with flexibility arising from the parameterized specifications.

These higher order objects can also be included in the library, allowing for mechanisms of ever increasing complexity. This allows a user to describe their desired robot in a more natural manner: rather than manually laying out edges and faces, a gripper module can be defined by attaching two fingers to a base, and connecting it to a servo module for actuation. The finger and servo modules were similarly modularly designed, and the gripper assembly can then itself be added to the library. Some elements currently implemented in the design library can be seen in Figure 1.

This modular paradigm has also been applied to the software required for controlling the robot and interfacing with its various components. Just as there are common mechanical or electrical building blocks that a user would want to use, there are common software snippets that the system can offer for control and interfaces; for example, a motor is usually controlled by setting its speed and a joint is usually controlled by setting its angle. Thus, such code can be written by expert users to augment the electromechanical modules with software and user interface (UI) elements. Furthermore, since the system is aware of the modular layout, it can decide how the devices should be wired to the central processor and therefore automatically generate code for the brain which can interface with the distributed electrical devices.

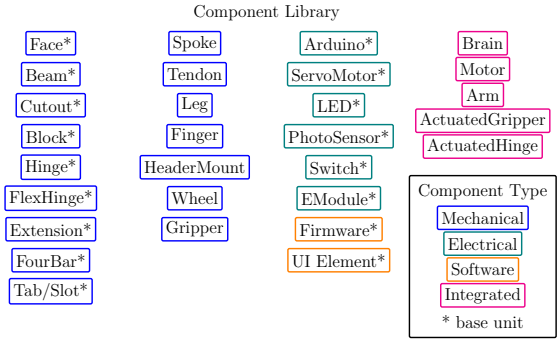

Fig. 1. A library of modular components enables robotic design to be reduced to hierarchical composition of pre-designed elements. The starred components are basic building blocks defined from scripts by experts; the rest have been assembled within the design system and added to the library.

A custom robot can therefore be designed, fabricated, and controlled by novice users utilizing these augmented modules. Once a high-level description of the modular layout is provided, the system can automatically compose the subcomponents stored within each module to generate all necessary instructions and files. It ensures that the mechanical elements are physically joined into the final structure, wires the electronic modules to the adjacent devices, and generates a final code package that can be programmed onto the central processor. The user then follows the generated instructions to build the robot, and can immediately control it using the automatically generated user interface.

IV. Co-design implementation

To implement the goals described in the preceding section, the script-based paradigm for designing mechanical components presented in [1], [2] was extended to include a method of modularizing distributed electronic systems and software. This forms the modular approach to defining complete robot designs by software code objects. Mechanical blocks, electrical blocks, integrated electromechanical modules, and software can be collected into a library for use in higher order designs.

A. Modularized mechanical structures

The modular design of structural elements using Python scripts is described in [2]. An origami inspired cut-and-fold fabrication process is used to generate 3D structures from 2D patterned sheets of plastic. A library of basic mechanical components form the building blocks from which complex geometries can be built by casual users, and then quickly fabricated using inexpensive tools and materials.

B. Modularized electronics

Typical robotic functionality includes various forms of sensing, actuation, processing, communication, and user interfacing. Many of the electromechanical transducers required to accomplish physical tasks are often distributed

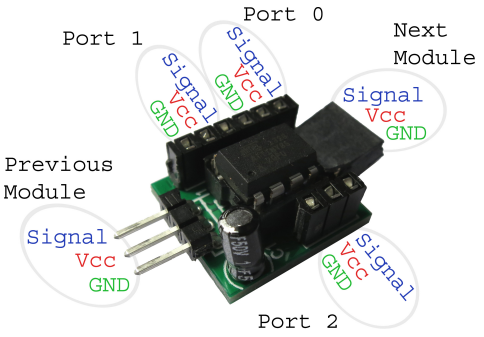

Fig. 2. Each electrical module features connections for an upstream and downstream module as well as three ports for connecting devices such as servos, LEDs, or digital and analog sensors. These modules are designed to be plug-and-play and do not require reprogramming based upon location or connected devices.

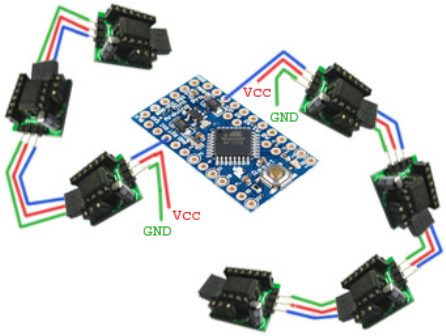

Fig. 3. Modules can be connected together to form distributed chains. In this case, two chains have been added to the brain and devices such as motors, LEDs, or sensors can then be plugged into any of the modules' ports.

throughout the robot, while other devices such as the core processor are collected on a central controller board. This generally requires the design and fabrication of a new controller board for each unique robot. Though techniques exist to generate such printed circuit boards (PCBs) [1], creating new printed circuit boards for each new robot requires a long design cycle that becomes expensive and untenable for casual users. A new system is therefore presented which allows electrical layouts to mirror distributed mechanical layouts, supports reusability, and allows novices to easily design complex systems.

The modularity hinges upon small plug-and-play modules that can communicate with each other and connect to devices such as actuators or sensors. The modules, shown in Figure 2, each feature an ATtiny85 microcontroller and three available ports to which devices can be connected. These ports can be used for driving general purpose digital outputs or pulse-width modulation (PWM) controls, and can measure digital or analog inputs. Modules can be connected together as chains as shown in Figure 3, in which each module communicates with its neighbors via a standard one-wire serial protocol. These chains are ultimately connected to a microcontroller unit acting as the brain of the robot. Most common sensors and actuators can be driven by these modules, thus minimizing the need for custom PCBs in most cases. For special purpose devices, a custom PCB need only provide the standard three-wire interface to integrate into any other designed system, allowing for general purpose reuse.

By utilizing this distributed configuration in which devices are attached to modules at the location of interest on the robot and information is sent along chains to control the devices or read from sensors, the robot's capabilities are no longer limited by the physical layout of a brain – as many modules as desired can be quickly added to the robot. Furthermore, the modules are exactly the same regardless of how they are used; the microcontrollers are programmed with code that does not need modification when using different robots or components. The brain is programmed with auto-generated code defining the particular configuration of the robot, and communicates the necessary information to the modules for configuring their pin types. In this way, leveraging pre-made plug-and-play modules promotes reusability and allows electrical layouts to match the distributed nature of mechanical configurations, so that software components can easily encapsulate electrical functionality along with structural elements.

This modularization facilitates the creation of basic reusable electrical blocks including sensors, LEDs, and servomotors. These can be assembled into common higher order compositions, such as joints, wheels, or grippers; the user can use the dimensions of the electrical components to appropriately adjust the exposed parameters of the mechanical components in order to ensure compatibility. Since the physical modules are generic and identical, the system can automatically decide an appropriate layout and thereby eliminate the need for users to consider appropriate pin usage, availability, or capability. It then outputs instructions for how to plug all of the devices together to realize the generated design.

C. Automatic code and UI generation

The library of electrical blocks can be augmented with driver code and UI elements for controlling the physical modules as well as their connected devices. As a result, the configuration of electrical modules can be processed to automatically generate C++ code for a microcontroller brain, such as an Arduino, which can configure all of the modules' pin types at startup and provide a framework for controlling all of the connected devices. Furthermore, the user can immediately control the robot via an auto-generated user interface; a smartphone application wirelessly communicates with the robot to determine its configuration, then compiles the UI elements from the software modules and presents appropriate controls for each attached device. Advanced users can also leverage the auto-generated code library to define custom interface controls better suited for their particular robot. A few lines of high-level descriptive code therefore allow a user with limited engineering experience to create software to control an arbitrary robot. An example of using this modularized electrical and UI system is shown in Figure 4.

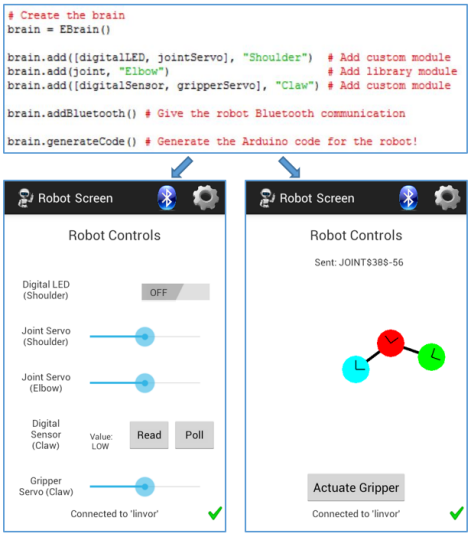

Fig. 4. The above code illustrates the definition of electronics for a robotic arm with two joints, one of which has an LED on it, and a gripper with a sensor. The arm can be controlled via the auto-generated UI on the left or a custom-written graphical interface on the right.

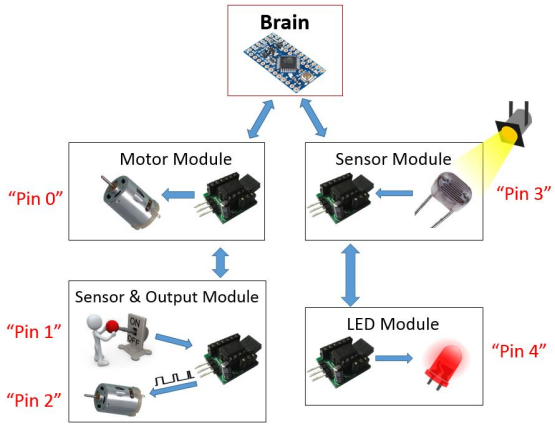

Fig. 5. As devices are added to the design, each one is automatically assigned a virtual pin number. Users can then control the robot using the virtual pin numbers, so that knowledge of the actual chain configuration is not required.

In addition to allowing novice users to control their robot, the generated code provides a framework for easily writing custom code. When the system analyzes the described robot configuration, it assigns each device a "virtual pin number," as shown in Figure 5, and this list of virtual pins is presented to the user along with the building instructions. If the user then opens the generated Arduino file, they can interface with the attached devices by simply using the virtual pin numbers. For example, if a sensor was assigned a virtual pin number of 3, a user can simply call robot.analogRead(3) as if it was connected directly to the brain. The generated robot library will determine the corresponding module and physical pin, and send the command along the appropriate chain. The user can therefore program as they normally would program an Arduino, and all of the work for interfacing with the actual electronic layout is done behind the scenes. Similar methods are also provided for using the Bluetooth to communicate with the smartphone and for common tasks such as setting a motor speed or joint angle.

D. Integrated library-based design

The components described above have been combined into an augmented library of electromechanical modules which also contain associated code and UI elements. This then provides the foundation from which users can design a complete robotic system, assembling intuitive building blocks into an integrated electromechanical robot together with its human interface.

In order to realize a custom robot to solve a particular task, a user determines how to break the task into simple subtasks, and identifies modules from the library to accomplish each subtask. If a specific desired item does not exist, an alternate similar one may be adapted, or a new module may be composed from lower level components. In the end, the description is compiled to generate the required design files; drawings can be sent to a cutter to fabricate the cut-and-fold mechanical structures and the generated software is programmed directly onto the brain module. Instructions on where to mount the electrical and electromechanical components are displayed, and the user then assembles the physical elements to create the final desired system and uses the auto-generated UI to control it.

V. Case Studies

This method was used to design and fabricate several different compiled robots, demonstrating a variety of distinct electromechanical mechanisms.

A. Two-Wheeled Roller

A simple Segway-inspired two-wheeled mobile robot is shown in Figure 7. The robot, nicknamed the Seg, is specified by three parameters:

- the dimensions of the brain module (in this case the Arduino Pro Mini),
- the dimensions of the continuous rotation servos used as drive motors (in this case Turnigy TGY-1370's), and
- the desired ground clearance (in this case 25mm).

Note that the dimensions of the chosen electrical devices are used to scale the chosen mechanical components (by adjusting parameters exposed by the designer of the mechanical blocks). The user can design a Seg using the enhanced electromechanical library by attaching two motor mounts to a central body, along with a tail for stability. The resulting component-based hierarchical design is presented in Figure 6.

Since the necessary mechanical and electrical components are encapsulated within the modules, the compiler creates design drawings for the body and wheels as well as code for the brain. Each motor mount provides information about the hardware as well as software and firmware. These get added to the core brain module, associated with the central body, as the mechanical mounts get physically linked. Instructions for connecting the modules and devices are then displayed to

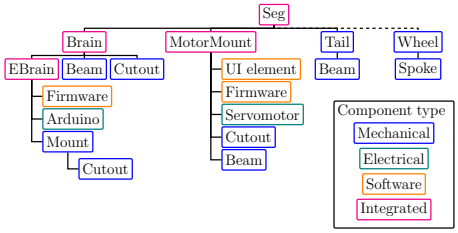

Fig. 6. Each node on this tree represents a component in the design of the two-wheeled robot, generated solely by composing its child nodes. The leaf nodes were design by expert designers, but every higher level of the design can be assembled from its children by a casual user.

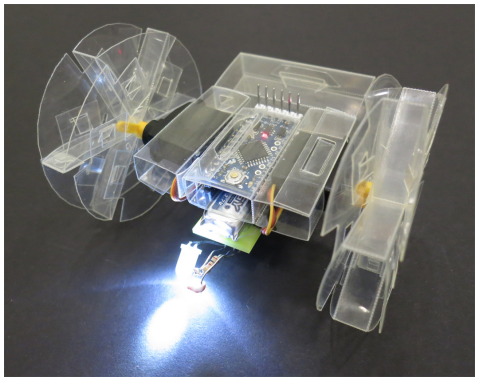

Fig. 7. The Seg, a two-wheeled mobile robot, was compiled from modular electromechanical components. Electrical components are directly connected to the brain using the modular software interface.

the user, and the app can immediately be used to control the robot. Auto-generated UI controls can be used to address each individual electronic component, or a virtual joystick can be implemented by an expert for a more intuitive controller. A summary of the robot's characteristics are provided in Table I.

This robot can also be used as a platform to explore custom software development. For example, the generated software library was employed to write a program for autonomous execution. Only a few simple lines of custom code were needed to use the light sensor to detect whether the robot is over black or white paper and thereby allow the robot to follow a line. The robot could then use provided methods for determining if the app is currently connected to switch between Bluetooth mode (controlled by the graphical interface), or autonomous mode (following black lines on the floor).

TABLE I

PERFORMANCE OF TWO-WHEELED SEG ROBOT

Metric	Result	
Approximate design time	1	hr
Approximate fabrication time	20	min
Approximate Cost	20.00	USD
Weight	42	g
Maximum speed	23	cm/s
Turning radius (both wheels activated)	0	cm
Turning radius (one wheel activated)	4	cm

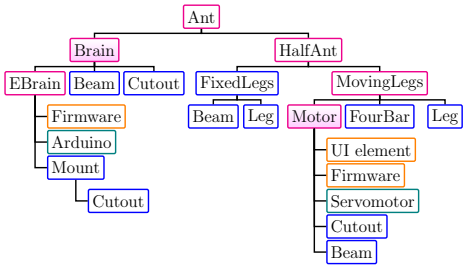

Fig. 8. The design of the walking robot is similar to that of the Seg, with the addition of mechanical leg and flexure components. The higher-level brain and motor components (shaded), can be reused from the earlier design.

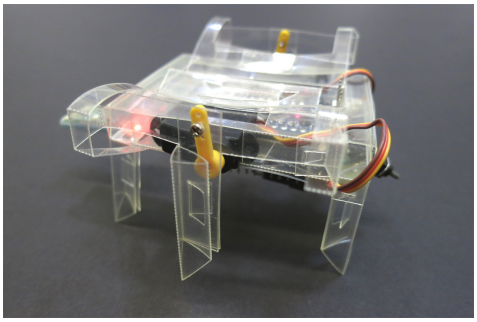

Fig. 9. A complex hexapod walker can be generated by adapting existing library elements generated from past designs.

B. Hexapod Walker

A more complicated mechanical structure can be employed to create a legged walking robot. In this case, the drive motors circularly actuate legs with two degrees of freedom, constrained to move in a plane by flexural four-bar linkages. Fixed legs provide a stable base. This demonstrates the versatility of modular design: components can be assembled hierarchically to define complex mechanisms with ease, with similar structures copied between components. The component hierarchy can be seen in Figure 8, and the resulting structure can be seen in Figure 9. The bulk of the design was taken from the Seg robot, thus greatly reducing design time.

C. Grasping Arm

In contrast to the locomotive robots above, an alternate configuration is presented by the multi-segment arm shown in Figure 11. In this case, the design specifications call for actuated joints to control the position of an end effector. The end effector itself is an integrated electromechanical mechanism included in the higher level assembly; the entire design hierarchy can be seen in Figure 10. This robot employs modular electronic components, with each actuated joint containing its own servo and drive circuits. The distributed nature of devices within the arm highlights the advantage of a distributed electrical system made possible by the plug-and-play electronic modules. This also allows for the daisy-chaining of arbitrarily many components, enabling robotic designs of ever-increasing complexity.

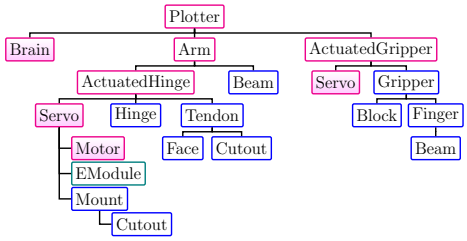

Fig. 10. The design tree for the gripper arm shows how a complex electromechanical device can be hierarchically assembled from simpler mechanisms. The integrated brain and servo modules are adapted from the earlier robots with slight modifications to enable daisy chained electronic modules, and the servo module is shared between the hinge and gripper mechanisms.

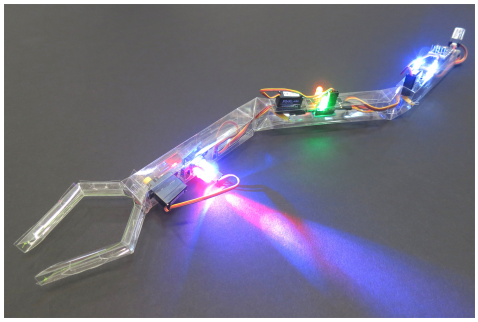

Fig. 11. A robotic arm was generated by connecting two actuated joints and an actuated gripper in the modular software interface.

Using the generated code, the arm could be immediately controlled using the automatically generated UI or the graphical arm depicted in Figure 4. Some performance metrics are also presented in Table II.

VI. Conclusions and Future Work

The presented system embodies a paradigm which rethinks the way in which robots are designed and constructed. By utilizing a modular system for both the mechanical and electrical components, a library of enhanced electromechanical modules with relevant software has been created. Experts can define new base components in a matter of hours, while casual users can define new integrated modules from base components in a matter of minutes. These modules can then be reused in new ways for different robots, and can be hierarchically combined to create arbitrarily complex designs using the provided script-based infrastructure. This modular encapsulation enables the system to automatically generate fabrication files, control software, and user interfaces based upon high-level structural descriptions. Three diverse robots generated with this system have been tested and presented, demonstrating that users with limited technical knowledge can quickly and inexpensively create robots. This system therefore makes strides towards the goal of a complete robot compiler that brings custom robotics into the domain of personal use and changes how robots are integrated into daily life.

TABLE II

Performance of Robotic Arm

Metric	Result	
Approximate design time	1	hr
Approximate fabrication time	30	min
Approximate cost	27.00	USD
Weight	60	g
Maximum joint angle (actuated)	± 35	deg
Maximum joint angle (mechanism)	± 110	deg
Gripper Strength (on 1.5 cm object)	100	mN

While the foundations for a robot compiler have been described and tested, future work must be done to effect a more complete integration of the various subsystems and provide a wider range of functionality. By further developing the provided libraries and forging a tighter connection between the mechanical and electrical designs, the necessary structural descriptions can be simplified and more diverse robots can be created. The system can also be augmented to actively aid the design process by helping the user determine a hierarchical composition, suggesting components, performing verification, and automatically resolving errors. Different fabrication methods, such as 3D printing, may also be incorporated to extend the functionality of the generated robots. Then, additional layers of abstraction can be added to allow definitions via functional specification rather than individually composing scripted components.

Acknowledgments

This work was funded in part by NSF grants 1240383 and 1138967 and NSF Graduate Research Fellowship 1122374, for which the authors express thanks.

References

[1] A. M. Mehta, D. Rus, K. Mohta, Y. Mulgaonkar, M. Piccoli, and V. Kumar, "A scripted printable quadrotor: Rapid design and fabrication of a folded MAV," in *16th International Symposium on Robotics Research*, 2013 (to appear).

[2] A. M. Mehta and D. Rus, "An end-to-end system for designing mechanical structures for print-and-fold robots," in *Robotics and Automation (ICRA), 2014 IEEE International Conference on*, 2014 (to appear).

[3] E. Hawkes *et al.*, "Programmable matter by folding," *Proceedings of the National Academy of Sciences*, vol. 107, no. 28, pp. 12 441–12 445, 2010.

[4] C. D. Onal, R. J. Wood, and D. Rus, "Towards printable robotics: Origami-inspired planar fabrication of three-dimensional mechanisms," in *Robotics and Automation (ICRA), 2011.* IEEE, 2011, pp. 4608–4613.

[5] P. Birkmeyer, K. Peterson, and R. S. Fearing, "Dash: A dynamic 16g hexapedal robot," in *Intelligent Robots and Systems (IROS), 2009.* IEEE, 2009, pp. 2683–2689.

[6] C. Onal, R. Wood, and D. Rus, "An origami-inspired approach to worm robots," *Mechatronics, IEEE/ASME Transactions on*, vol. 18, no. 2, pp. 430–438, April 2013.

[7] I. Shimoyama, H. Miura, K. Suzuki, and Y. Ezura, "Insect-like microrobots with external skeletons," *Control Systems, IEEE*, vol. 13, no. 1, pp. 37–41, 1993.

[8] A. M. Hoover and R. S. Fearing, "Fast scale prototyping for folded millirobots," in *Robotics and Automation (ICRA), 2008.* IEEE, 2008, pp. 886–892.

[9] E. D. Demaine and T. Tachi, "Origamizer: A practical algorithm for folding any polyhedron," 2009.

[10] R. Lang, *Origami design secrets : mathematical methods for an ancient art*. A K Peters/CRC Press, 2012.

[11] "Pepakura designer," http://www.tamasoft.co.jp/pepakura-en/, [Online; accessed 01-Feb-2014].