roBlocks: A Robotic Construction Kit for Mathematics and Science Education

Eric Schweikardt
Carnegie Mellon University
tza@cmu.edu

Mark D. Gross
Carnegie Mellon University
mdgross@cmu.edu

ABSTRACT
We describe work in progress on roBlocks, a computational construction kit that encourages users to experiment and play with a collection of sensor, logic and actuator blocks, exposing them to a variety of advanced concepts including kinematics, feedback and distributed control. Its interface presents novice users with a simple, tangible set of robotic blocks, whereas advanced users work with software tools to analyze and rewrite the programs embedded in each block. Early results suggest that roBlocks may be an effective vehicle to expose young people to complex ideas in science, technology, engineering and mathematics.

Categories and Subject Descriptors
K.3.2 [**Computer Science Education**]

General Terms
Algorithms, Design, Human Factors, Languages.

Keywords
Robotics education, Construction kit, Tangible interface

1. INTRODUCTION
Evidence, both anecdotal and research-based, suggests that playing with construction kits like LEGO, Erector Sets, or Lincoln Logs as a child can foster creativity, math skills and mechanical aptitude. Frank Lloyd Wright, who credits the geometric blocks made by Friedrich Froebel for strongly influencing his career [5] is not alone in giving credit to construction kits for supporting creativity and encouraging design. Recently, many researchers have made the case that design projects present an effective framework for learning, [4] and the expanding availability of commercial construction kits demonstrates their popularity.

Today, with more technology available in smaller sizes and at lower cost, there is room for next-generation construction kits to be built for young inventors. Existing computationally-enhanced construction kits, such as LEGO Mindstorms or the VEX Robotics Design System, combine building sets with one small computer that executes instructions, often in the C language, and centrally controls motors and simple sensors. While these kits can be used to create exciting dynamic constructions, embedding computation throughout the kit instead of in one central location creates rich opportunities for augmenting a user's experience.

Robotics provides a compelling domain for experimenting with the design of complex systems. Even with current kits, however, actually constructing robots that exhibit interesting behaviors usually involves a high degree of technical experience and skill in several domains: mechanics, electronics, and programming. We describe roBlocks, a computational construction kit designed to scaffold children's math, science, and engineering education.

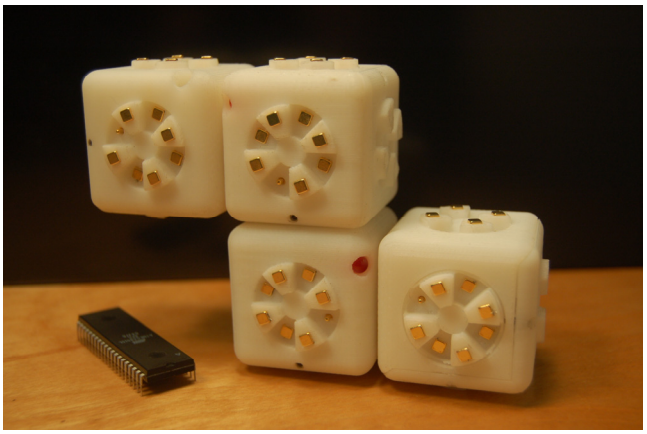

Figure 1. A simple roBlocks construction.

In addition to covering the world of static constructions, this research aims to explore children's conceptions of logic, cause-and-effect, feedback, kinematics, distributed systems, and allow them to "gain a deeper understanding of how dynamic systems behave." [12] The roBlocks system, comprised of a kit of robotic blocks and a software package, encourages users to program by connecting blocks, analyze their constructions via on-screen tools, and eventually reprogram their creations.

We intend the basic roBlocks functionality to be accessible to children as young as nine, who will be able to snap together blocks and create simple robots, observing the actions of the construction and inferring how the rules of the system combine to create behaviors. In addition, we intend a more advanced deployment of roBlocks for use in undergraduate education, providing a robust set of primitives for experimentation with high-level concepts. In many undergraduate robotics courses, such as Carnegie Mellon's *General Robotics* or MIT's 6.270, low-level details like construction with LEGO and electronics work can overwhelm many students, distracting them from gaining a full understanding of concepts like distributed control or closed- and open-loop feedback [9]. We intend roBlocks to expose these fundamental control theory ideas.

ICMI '06, November 2–4, 2006, Banff, Alberta, Canada.

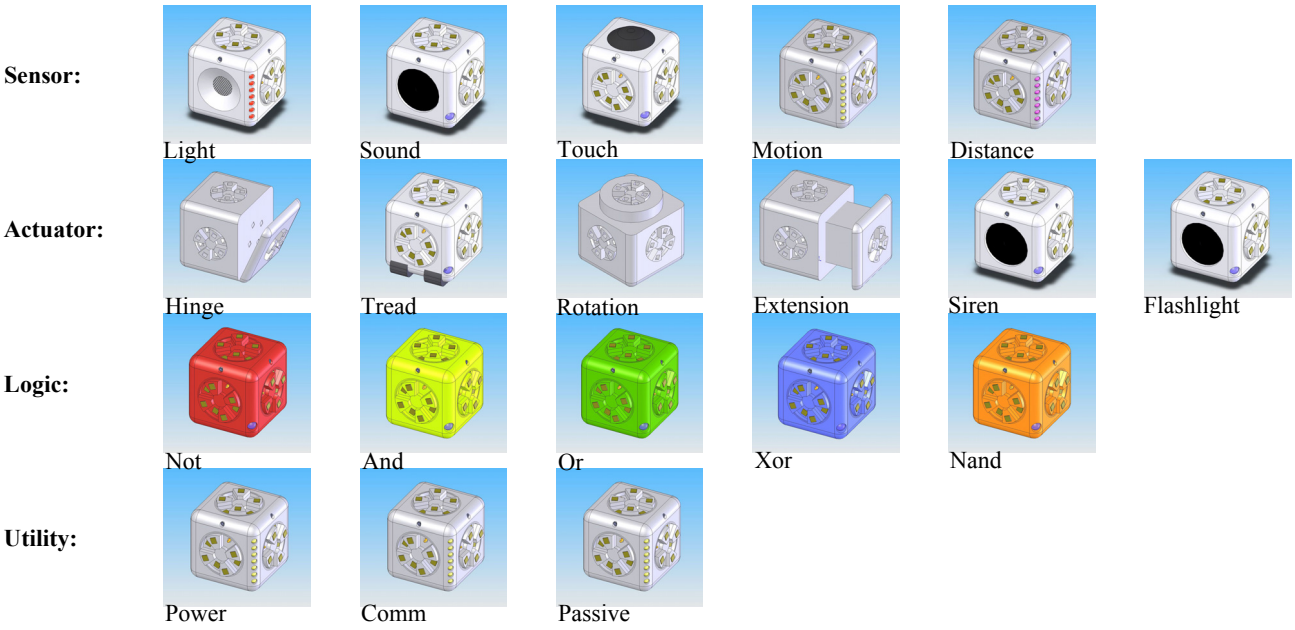

Figure 2. The roBlocks catalog.

2. ROBLOCKS HARDWARE

roBlocks are 40mm plastic cubes with magnetic connectors on their faces. The connectors provide both physical and electrical connection, and are hermaphroditic, so that any block face may connect to any face of another block. Each block contains a microprocessor and is addressable as a node on the ad-hoc network created as roBlocks are connected. Nineteen different roBlocks are divided into four categories: Sensors, Actuators, Logic, and Utility blocks. The behavior of each roBlock is apparent from its appearance.

We print each roBlock body on a Fused Deposition Modeler in two halves. One half, with three faces and their connectors, contains the roBlocks base electronics and snaps onto the second half, which is specific to each block. Currently, we painstakingly assemble each roBlock by hand.

2.1 Sensor Blocks

Providing input to robotic constructions, each Sensor block includes a specific transducer to allow a roBlock construction to respond to real-world stimulus or conditions. The five sensor blocks are Light, Sound, Touch, Motion and Distance. Sensor blocks are continuously active when connected to a construction, and feature a simple bar-graph LED readout of relative sensor values on one edge of the block. The Light roBlock, for instance, includes a photocell resistor on one of its faces (instead of the usual connector). When connected to a construction, the block is powered on, and the block's microprocessor converts its analog sensor data into numeric values, and broadcasts the data over the block network. The LED display helps the user visualize the sensor values and correlate the current conditions with what the roBlock "sees." Our group has built a Light block (using a passive photoresistor) and a Distance block (using a Sharp IR rangefinder), and is working to refine them and construct the rest.

2.2 Actuator Blocks

The actuator blocks are the most exciting blocks for many users, creating motion, light and sound, and bringing a construction "to life." The Hinge, Belt, Extension and Rotation blocks each contain a small gear motor and add a degree of freedom to a construction. There is also a Sound block with a piezoelectric speaker and a Flashlight block with a bright white LED and a focused reflector. We have built working versions of the Rotation and Flashlight blocks.

2.3 Logic Blocks

The five logic blocks (And, Or, Not, Nand, Xor) act as tangible programming statements, allowing users to create simple programs just by snapping individual roBlocks together. Adding a Not block between a Sensor and Actuator block would cause the actuator to operate on an inverse relationship, instead of directly. Adding an And block between two sensors would cause an Actuator block to operate only when both sensors are activated.

2.4 Utility Blocks

There are three Utility blocks. The Power block contains a rechargeable NiMH battery and power supply circuit, and a master switch to turn the robot on or off. Every robot must contain a Power block, which provides power to the rest of the blocks via a power/ground bus routed through magnetic connectors. The Comm block enables a robotic construction to communicate with a PC via an embedded Zigbee wireless transceiver. A USB port is also provided, both for communication (wireless is disabled when the robot is plugged in) and to charge any connected Power blocks. The Passive roBlock is simply a roBlock with no additional functionality but still includes a microprocessor and unique address.

3. SOFTWARE

Each roBlock contains an Atmel ATmega645 AVR microcontroller preprogrammed with a series of behavioral rules. A roBlock construction is a distributed system; there is no overarching program controlling the individual blocks. At level one, users build constructions that operate based on the block's default programming. Level two exposes users to the programs running on each block and alters the programs based on construction topology, but doesn't allow users to modify them directly. Level three opens the roBlocks software architecture to allow advanced and/or older users, to reprogram individual blocks and experiment with the control of distributed systems.

4. INTERACTION

In order to be both accessible to young children and still support more experienced users, the roBlocks interface becomes more complex and powerful over three levels. Users begin simply by building with physical cubes, later adding on-screen display and manipulation, and finally adding custom programming to their creations.

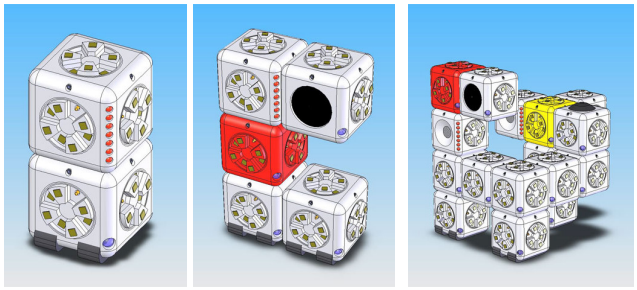

Figure 3. Three Possible roBlock constructions.

4.1 Level One

For novice users, the interface consists solely of blocks: children connect roBlocks to create robots that respond to sensor input with actuator blocks linked via logic blocks. The entire robot is built simply by snapping together the block primitives; no additional programming or orchestration is needed. At this level, all programming is hidden from the user; it is embedded in the physical pieces used to construct the robot. Accordingly, a construction contains no internal representation of its topology. Therefore users are free to build buggy robots, for example to connect opposing actuators that will break apart the construction when activated.

Although Level One is the most basic level of interaction, interesting robots can certainly be constructed. The robot on the left side of Figure 3 shows a Light sensor block connected to a Tread actuator block to create a robot that drives toward a light source. The middle robot adds a Not logic block between them, and also a Siren block, creating a robot that makes noise and turns away from any light source. On the right, a more complicated construction drives toward a light source, makes noise when it's dark, and wags its tail when there's a light on and somebody touches it.

4.2 Level Two

For a richer experience, a Comm block can attach to a roBlock construction in order to communicate with a host PC running the roBlocks Desktop software. An isometric view of the robot is rendered on-screen, and is updated in real-time as changes are made to the physical robot. A graphic display of sensor data is also shown. At this point, the user can manipulate an actuator on the physical robot and see the screen image change; conversely, drag an on-screen block and watch the physical robot emulate the on-screen action. Blocks moved on-screen are constrained by their physical construction, which prevents a user from directing the robot to break itself.

In addition, the Comm block queries all the blocks in the construction to create an internal representation of topology. At this point, knowing how it's been constructed, the robot can alter its programs to optimize behavior. The construction in Level One that destroyed itself now drives its actuators in tandem. A construction with two actuators and one sensor now powers its wheels as a differential drive to steer toward the input, instead of simply powering the motors in tandem. The Comm block, by coordinating and communicating with each block in the assembly, serves in effect as a brain, transforming a construction from a network of local actions into a network with global awareness.

4.3 Level Three

Level Three finally adds explicit programming to the user's experience. Moving the mouse over an individual block on the screen causes a small window containing the block's main program to appear. The user is free to modify the block's program, adding explicit relationships to other blocks, conditional statements, or any other construct they desire. We are modeling the roBlocks programming interface on Alice [10] or Scratch [6], which feature simple syntax and graphically nested loop and conditional statements. A bright green LED on one corner of each roBlock lights when the block is reprogrammed, alerting the user that its default functionality has been modified.

5. LEARNING OUTCOMES

This work is motivated by the idea that experimentation and play with robots exposes students to many subject areas within science, math and engineering. Seymour Papert's idea of "soap-sculpture math," in which math classes begin to embody some of the creativity, excitement and ownership traditionally present in art studios, [9] compels us to continue developing roBlocks so we can test it in classrooms.

Another motivator for this work is a desire to enable more students to study and participate in the field of robotics. We believe that robotics, due to its inherent complexity, has become somewhat insular, and could benefit from increased interdisciplinary collaboration with designers, materials scientists, psychologists, and other creative people. We note that the grand visions of robotic helpers from the 1950s have, by and large, been transformed into the reality of only one popular robotic product, a vacuum cleaner.

An important educational benefit of exposing younger children to robotic systems is the opportunity for them to begin understanding different theories of control. A recent study conducted in Israel, [8] showed that children's ideas about robotic control can develop quickly. Starting out with anthropomorphic concepts like "the robot seems hungry," kindergarten students were able to describe more accurate procedural rule-sets after

experimenting with robot kits. Early exposure to these concepts may be valuable, eliminating the need to re-learn these concepts later. In discussing the undergraduate robot competitions at MIT, Fred Martin [7] describes several failures, including the students' tendency to take an "omniscient" perspective when designing their robots. Earlier exposure to local control and negative feedback techniques might spur students to work "from the robot's point of view." The decentralized structure of roBlocks allows them to function like Braitenberg's Vehicles [2], embodying Brooks's "intelligence without representation" [3].

6. RELATED WORK

The roBlocks project builds on previous work in several disciplines. MERL's *Computational Building Blocks[1]* are self-describing bricks inspired by LEGO. They communicate asynchronously and contain microcontrollers in each brick, but are limited to static constructions. *Electronic Blocks* are [14] tangible programming elements that allow children to create simple robotic experiments. The sensor, logic and action blocks that comprise the system are large plastic blocks that rely on simple stacking to create a program. MIT's *System Blocks* [15] are tangible blocks that model system dynamics but they are limited to audio output.

The ActiveCube project at Osaka University [13] is an interface system with a large catalog of plastic blocks and magnetic connectors. The self-describing blocks contain accelerometers to detect their orientation, and are intended to act as a tangible interface to a virtual environment. *Topobo*, [11] at MIT's Media Lab, is an interesting example of programming robotic structures by example. Children teach their construction simple motions by twisting bricks, but the system works only with local rules, unaware of its configuration.

7. FINAL REMARKS

The current roBlocks hardware needs further development before testing with users but anecdotal evidence demonstrates that roBlocks will engage users as young as six. We spent a significant amount of time refining the physical design so that children and adults would enjoy playing with roBlocks and feel compelled to experiment with robotics. Early tests show that most users find them appealing: they are excited to manipulate the blocks, snapping them together in different configurations and observing their behavior.

We are currently refining the hardware design so that we can build roBlocks more quickly. We have begun working on a hardware version using printed circuit boards directly attached to the magnetic connectors with conductive epoxy, and we are also developing the mechanics for the rest of the actuator blocks. We have no convictions about the completeness of the roBlocks kit and thus we are entertaining ideas for blocks that support other forms of sensing, actuating, and communication. In addition, we are building the interface to Level Three, and prototyping different programming environments to determine what methods will be effective for reprogramming the blocks in Level Three.

8. ACKNOWLEDGMENTS

We thank the anonymous ICMI reviewers for helpful feedback. This research was supported in part by the National Science Foundation under Grant ITR-0326054. The views and findings contained in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

9. REFERENCES

[1] Anderson, D., Frankel, J., Marks, J., Agarwala, A., Beardsley, P., Hodgins, J., Leigh, D., Ryall, K., Sullivan, E., Yedidia, J. Tangible Interaction + Graphical Interpretation: A New Approach to 3D Modeling. in *SIGGRAPH 2000*, ACM, 2000, 393-402.

[2] Braitenberg, V. *Vehicles: Experiments in Synthetic Psychology*. MIT Press, Cambridge, MA, 1984.

[3] Brooks, R. Intelligence without Representation. *Artificial Intelligence*, *47*. 139-159.

[4] Brosterman, N. *Inventing Kindergarten*. Abrams, New York, 1997.

[5] Kaufmann, E.J. *Nine Commentaries on Frank Lloyd Wright*. MIT Press, Cambridge, MA, 1990.

[6] Maloney, J., Burd, L., Resnick, M., Silverman, B., Rusk, N. and Kafai, Y. Scratch: a Sneak Preview. in *Second International Conference on Creating, Connecting, and Collaborating through Computing,*, Kyoto, 2004, 104-109.

[7] Martin, F. Ideal and real systems: A study of notions of control in undergraduates who design robots. in Kafai, Y. and Resnick, M. eds. *Constructionism in Practice*, Erlbaum, Mahwah, NJ, 1994.

[8] Mioduser, D., Levy, S.T. and Talis, V. Kindergarten children's perception of robotic-control rules. in *Intl Conf Learning Sciences* Seattle WA, 2002.

[9] Papert, S. Situating Constructionism. in Harel, I. and Papert, S. eds. *Constructionism*, Ablex Publishing Company, Norwood, NJ, 1991, 1-11.

[10] Pausch, R., Burnette, T., Capehart, A.C., Conway, M., Cosgrove, D., DeLine, R., Durbin, J., Gossweiler, R., Koga, S. and White, J. Alice: A Rapid Prototyping System for 3D Graphics,. *IEEE Computer Graphics and Applications*, *15* (3). 8-11.

[11] Raffle, H., Parkes, A. and Ishii, H. Topobo: A constructive assembly system with kinetic memory. in *Human Factors in Computing (CHI) '04*, ACM, 2004.

[12] Resnick, M., Martin, F., Berg, R., Borovoy, R., Colella, V., Kramer, K. and Silverman, B., Digital manipulatives: new toys to think with. in *SIGCHI conference on Human factors in computing systems*, (Los Angeles, California, United States, 1998), 281-287.

[13] Watanabe, R., Itoh, Y., Asai, M., Kitamura, Y., Kishino, F. and Kikuchi, H. The Soul of ActiveCube - Implementing a Flexible, Multimodal, Three-Dimensional Spatial Tangible Interface. in *Proc. of ACM SIGCHI International Conference on Advanced Computer Entertainment Technology ACE 2004*, 2004, 173-180.

[14] Wyeth, P. and Wyeth, G. Electronic Blocks: Tangible Programming Elements for Preschoolers, in *INTERACT 2001*, 2001.

[15] Zuckerman, O. and Resnick, M. System Blocks: A Physical Interface for System Dynamics Simulation. in *Proc. of CHI (Computer-Human Interaction)'03 Conference*, 2003, 810-811