Computation and Construction Kits: Toward the Next Generation of Tangible Building Media for Children

Michael Eisenberg, Leah Buechley, Nwanua Elumeze
Department of Computer Science
University of Colorado, Boulder CO USA 80309-0430

ABSTRACT

Construction kits represent a venerable, creative, and (occasionally) even beautiful genre of educational toys for children. Nonetheless, traditional construction kits have limitations as educational media. In the past decade, a number of research efforts have attempted to address these limitations by augmenting construction kit design with various types of computational media. This paper describes two new prototypes of "computationally-enhanced construction kits"; unlike previous efforts, these newer kits allow for large numbers of mutually-interacting, computationally complex, geometrically innovative, and user-programmable pieces. We describe the current (still-early) state of these systems, and discuss plausible directions for future development and research.

Keywords
Construction kits, educational technology, embedded computing.

INTRODUCTION

In the history of educational design, construction kits (i.e., building toys comprised of multiple pieces) occupy an important and respectable place. Such toys date back at least to the eighteenth-century philosopher John Locke and his set of alphabet blocks ("Locke's blocks" [7, p. 119]), and the tradition has continued in a variety of guises through the nineteenth century and into the present day. (Harley [4] offers a brief summary of this tradition; while Watson [8] presents a thoughtful biography of A.C. Gilbert, the most prominent and inventive American designer of such toys in the twentieth century.)

Construction kits, viewed "in the large" as a genre of educational design, have marvelous strengths. They offer children a tangible medium for the creation of models in domains such as geometry (e.g., Zometool [PW2]), engineering (e.g., Erector sets [8]), and architecture, among others. (Widening our view a bit, we might also classify chemical modeling sets as relatively advanced construction kits that provide chemistry students with an indispensable aid to visualization; and likewise, modeling kits–such as anatomical kits–could be included in this genre as well.) By engaging children's visual and tactile senses, construction kits exhibit affordances that other sorts of children's artifacts (notably, video games) cannot match: a construction can be handed around from one child to another, or kept as a souvenir, or placed on a shelf for long-term viewing or exhibition, or built in such a way that it occupies a compellingly large area of a child's room. Nonetheless, there are, at the same time, profound limitations of "traditional" construction kits when viewed in the light of a more recent tradition of educational computing. The objects produced by traditional kits are typically static, or at most capable of a very limited range of behavior; as a consequence, traditional constructions run the risk of losing a child's interest after a short time. At the same time, the absence of complex behavior implies that traditional kits exhibit intellectual, content-related limitations as well. For instance, a molecular model cannot display to a student which particular atoms in a molecule of water have a slight excess of positive charge; an anatomical model cannot show a student the dynamics of neural signals between brain and hand; a geometrical model of an icosahedron cannot display which vertices are equivalent under a particular set of symmetry operations. (For a more extensive discussion, see [1].)

In recent years, a number of research efforts have augmented the design of construction kits with computational capabilities of various sorts. The most influential and prominent such example is the "programmable Lego brick" created by Resnick and his colleagues at the MIT Media Lab [6]. The brick can be programmed in Logo and can be combined with a variety of sensors and actuators to make dynamic Lego constructions; and it has given rise to a highly successful (though somewhat less versatile) commercial product, Lego Mindstorms. Other projects include the design of the ActiveCube [5], a set of cubical building blocks with embedded computers that can be combined into three-dimensional structures that can then be communicated to a desktop computer; "tangible programming" blocks created by P. Wyeth and G. Wyeth [9], a set of blocks that represent simple computational operations and that can be combined into (e.g.) wheeled vehicles with interesting behaviors; and a set of mutually-communicating triangular tiles created by Gorbet, Orth, and Ishii [3].

This paper describes two recent prototype systems, developed in our laboratory, that (in combination) explore a variety of novel directions in the creation of computationally-enhanced construction kits. In both cases, these kits can be seen as tangible media for the construction of cellular automata (a brief definition of cellular automata will be presented shortly). More pertinently for this discussion, however, the two kits represent a certain style of "computationally-enhanced construction kit" design that includes elements of user programmability (like the programmable Lego brick); large numbers of intercommunicating pieces (like the Active Cube and tangible programming block examples); and geometric innovation (like the triangular tiles). As we will discuss, neither kit, in its current instantiation, individually encompasses all these elements; but between them, they illustrate a style of design that eventually should permit more complex behaviors than those of (e.g.) the tangible programming blocks, larger sets of computing elements than are typical of programmable brick projects, and greater user customizability than the ActiveCube or triangular tiles.

The following section begins by presenting a brief summary description of cellular automata; in the third section, we explain the implementation and behavior of the two prototype systems. The fourth and concluding section describes plausible directions for future development of both prototypes, and discusses research directions for the design of next-generation computationally-enhanced construction kits.

CELLULAR AUTOMATA (IN BRIEF)

Before introducing our two prototypes, it is worthwhile to describe (as telegraphically as we can manage) the basic ideas behind cellular automata. Cellular automata are large collections of regularly-arranged simple computers; for example, the individual computers could be arranged in a square grid with each computer occupying a single square (or "cell"). Each computer communicates with a specified set of local neighbors (in our "typical" square-grid case, each computer might communicate only with its eight surrounding neighbors on the grid), and the cells generally operate in synchronized fashion, each running a simple program that operates from a global clock signal.

The most famous instance of a cellular automaton program is the celebrated "Game of Life" system, invented by John Conway and discussed in several delightful essays by Martin Gardner. [2] In the Game of Life, cells are arranged on a square grid; each cell communicates only with its eight surrounding neighbors; each cell has a simple one-bit state value (0 or 1, which can be thought of as "not alive" and "alive"); and each cell runs a simple program in which a cell at time T is alive at time T+1 if it is surrounded by exactly three surrounding live neighbors at time T, or if it is currently alive and surrounded by two live neighbors at time T. (Otherwise, the cell is not alive at time T+1.) Space does not permit a discussion of the marvelous properties of this simply-described system; for more detail, see the references cited above.

There are many possible variants on this basic description. In general, cellular automata are arranged in a plane (such as the Game of Life square grid), but this needn't always be the case, as we shall illustrate. Likewise, some models communicate in asynchronous fashion; some make use of far more complex programs (and far more elaborate internal state descriptions) than the Life example above; and so forth. For the purposes of this paper, however, the crucial point is that cellular automata illustrate a kind of "construction kit" of computational elements: large numbers of geometrically arranged, mutually-communicating computers are capable of fascinating collective behaviors. Our two prototypes make use of this aspect of cellular automata, as both permit the user to create cellular automaton models out of tangible pieces.

TWO PROTOTYPE CELLULAR-AUTOMATON CONSTRUCTION KITS

In this section, we describe our two prototype systems; though we cannot include extensive details about the systems here, we will concentrate on those aspects relevant to the larger subject of construction kit design.

Smart Tiles

Smart Tiles may be thought of as computational elements that can be programmed by the user and arranged into a plane, each tile communicating with its immediate neighbors. Figure 1 provides a view of a four-by-four set of tiles. In our current prototype, each tile is associated with a circular light that reflects its current state ("lit" versus "unlit"); and the tiles, once programmed, are placed on a fabric that acts as the background surface for the tiles, and that contains connections for power and communications. Each tile is able to communicate with its eight surrounding neighbors in the fabric, and each tile's individual program dictates the state of the tile (lit/unlit) over time in accordance with programmed rules that can access the tile's own state and that of its neighbors.

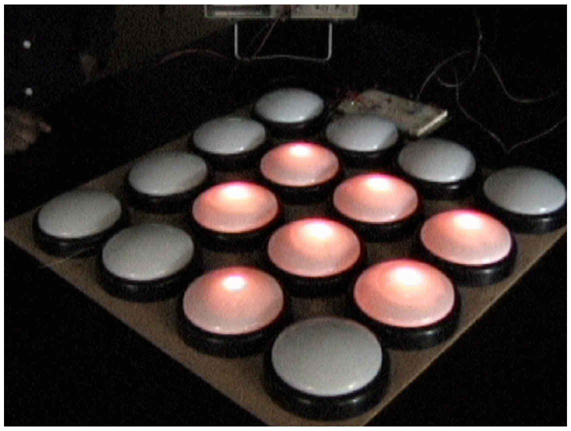
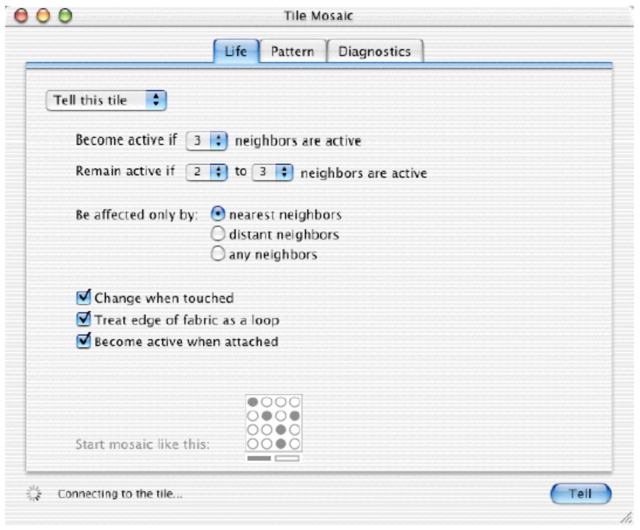


Figure 1. A four-by-four array of Smart Tiles.

In essence, then, each tile can be thought of as a single individually-programmable "piece" in a larger array of pieces that can be assembled into an array form, as shown in Figure 1. In the scenario shown in the figure, we happen to have programmed the tiles to act according to the Game of Life rules given above; thus, all tiles are running the same program. Importantly, though, it is possible to remove a single tile from the fabric, connect it to a desktop machine, and reprogram it so that it can run its own particular rule; the tile may then be reinserted into the fabric, from which point on it will run its own customized rule. The programming interface for the tiles is currently rather simple, and is shown in Figure 2; essentially, the user may choose (in menu-driven fashion) among certain "Life"-type rules. (A much more expressive tile-programming interface is currently in the works.) One other feature of the Smart Tiles system deserves mention here: namely, that tiles may have their state changed interactively, while a simulation is running, by pressing on a tile (this effectively toggles the current lit-versus-unlit state of the tile in the simulation). Thus, users may "play" interactively with an automaton program as it runs.



Tell this tile

Become active if 3 neighbors are active

Remain active if 2 to 3 neighbors are active

Be affected only by: [✓ nearest neighbors] distant neighbors any neighbors

☑ Change when touched
☑ Treat edge of fabric as a loop
☑ Become active when attached

Start mosaic like this:

● ○ ○ ○
○ ● ○ ●
○ ○ ● ○
○ ○ ● ○

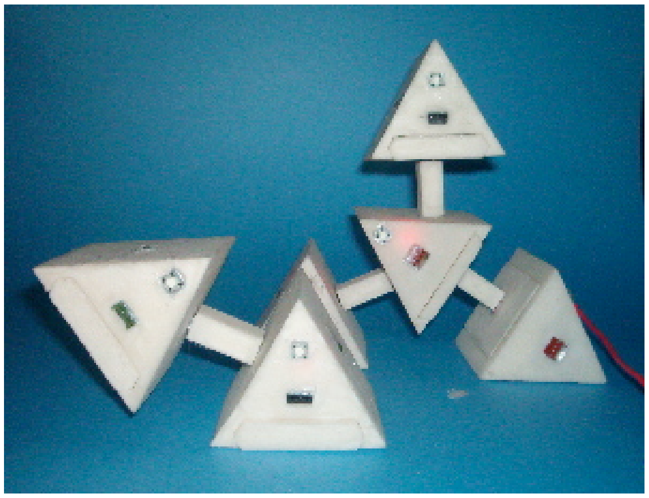
Connecting to the tile...

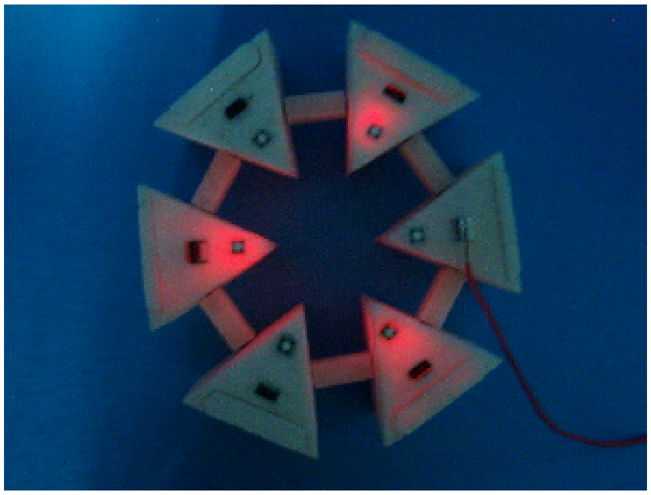
Figure 2. The Smart Tile programming interface.

Triangular Prism Automaton Kit

The Smart Tiles system described above can be used to create planar cellular automata in which pieces are arranged on a square grid. In contrast, the Triangular Prism Kit allows a user to build three-dimensional cellular automata. Here, the "pieces" are computational cells, each in the shape of a triangular prism; the pieces are connected (both physically and electronically) via struts. Figure 3 shows a sample construction built from the prisms; any such construction can be disconnected, and the pieces rearranged into myriad three-dimensional structures (or, on occasion, two-dimensional structures: Figure 4, for instance, shows six prisms arranged in a planar ring). A single designated "global" piece (visible toward the right of Figures 3 and 4) provides the overall construction with power and with a synchronizing clock signal; it thus plays the role of the "background fabric" in the Smart Tiles prototype. Again, in standard cellular-automaton fashion, each individual piece runs its own program, setting a state value (represented by a lit or unlit red light) in accordance with rules that depend on the cell's own state and that of its directly-connected neighbors.

Figure 4 illustrates a particular automaton rule for our triangular prisms: here, each prism is executing a rule that tells it to light up at time T+1 if and only if it has exactly two lit neighbors at time T. In the figure, three of the cells are lit; at the subsequent time step, these three cells will go dark, while the other three cells (in accordance with the stated rule) will light up. The overall ring will thus oscillate between two configurations, with alternating sets of three prisms lighting up and going dark at each time-step.





Figures 3 (above) and 4. Two running Triangular Prism constructions.

Both prototypes–Smart Tiles and Triangular Prisms–are implemented by using PIC microcontrollers as embedded computers within individual pieces. Currently, the user programming interface for the prisms is far more primitive than that for the tiles; the prisms are programmed before assembly in Jal, a Pascal-like language created by W. van Ooijen.[PW1] Likewise, the interactivity of the prisms is less advanced than that of the tiles: essentially, the user first sets up the initial state of a running system by setting switches on the prisms, after which the overall automaton construction begins running.

Future Directions for Development and Research

Both prototypes described here can be regarded as works-in-progress, and both are the subjects of continuing development. In the case of Smart Tiles, we have already largely implemented a second, much smaller version of the tile (see Figure 5), and we are working (as noted earlier) on a new generation of programming interface. In the case of the prisms, one direction for future development involves experimentation with different geometric forms for the pieces. Figure 6, for instance, shows a (non-working) model for pieces in the form of rhombic dodecahedra; these could be assembled into structures different from those of the prisms.

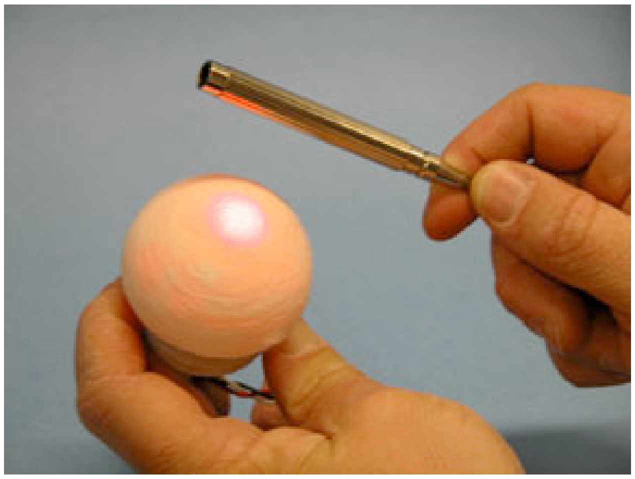


Figure 5. A smaller Smart Tile prototype (a "toadstool"). The figure shows the user toggling the tile's state by tapping on its surface; the tile has been made sensitive to this sort of input by the inclusion of a piezoelectric disk underneath the surface.

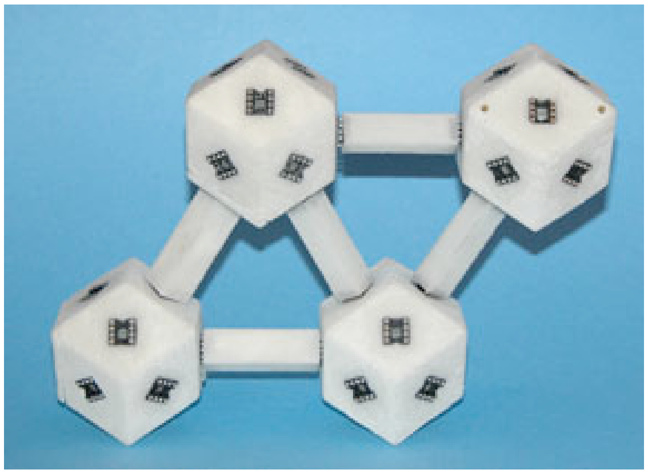


Figure 6. A construction kit design based on rhombic dodecahedra.

In addition to continued technical development, both prototypes have yet to be pilot-tested with children (or even with adult users!). Our first such tests (employing the tiles) are planned for early fall. Our initial focus in these tests will be to see whether students can understand the behavior of overall constructions; whether they can make use of a programming interface to customize their constructions in interesting or meaningful ways; and how (and whether) they interact with their constructions (e.g., by toggling tiles during a running simulation). While these projects are undeniably still at a relatively early stage, then, we believe that they afford fertile ground for the development of innovative, behaviorally complex construction kits; and we intend to use these systems as exemplars for our own continuing development of still other construction kits focusing on a variety of specific scientific and mathematical domains.

Acknowledgments

This work was supported by NSF ROLE grant EIA-0326054.

References

1. Eisenberg, M., et al. [2002] Computationally-Enhanced Construction Kits for Children: Prototype and Principles. In *Proceedings of ICLS (International Conference on the Learning Sciences)*, Seattle, WA.
2. Gardner, M. [1985] *Wheels, Life, and other Mathematical Amusements.* NY: W. H. Freeman.
3. Gorbet, M.G. et al. [1998] Triangles: Tangible Interfaces for Manipulation and Exploration of Digital Information Topography. In *Proceedings of CHI '98*, pp. 49-56.
4. Harley, B. [1990] *Constructional Toys*. Shire Publications: UK.
5. Kitamura, Y. et al. [2001] Real-time 3D Interaction with ActiveCube. *In Proceedings of CHI 2001* (Extended Abstracts), Seattle, WA, pp. 355-356.
6. Resnick, M. et al. [1996] Programmable Bricks: Toys to Think With.. *IBM Systems Journal*, 35:3, pp. 443-452.
7. Sutton-Smith, B. [1986] *Toys as Culture*. NY: Gardner Press.
8. Watson, B. [2002] *The Man Who Changed How Boys and Toys Were Made*. Viking Press.
9. Wyeth, P. and Wyeth, G. [2001] Electronic Blocks: Tangible Programming Elements for Preschoolers. In *Proceedings of the Eighth IFIP TC13 Conference on Human-Computer Interaction* (INTERACT 2001).

Products and Websites:

[PW1] Jal website: http://jal.sourceforge.net/

[PW2] *Zometool*. Zometool, Inc. (www.zometool.com)