Computers and Physical Construction: Blending Fabrication into Computer Science Education.

Conference Paper · January 2008
Source: DBLP

CITATION
1

READS
340

4 authors, including:

Michael Eisenberg
University of Colorado Boulder
88 PUBLICATIONS 1,661 CITATIONS

Ann Eisenberg
University of Colorado Boulder
15 PUBLICATIONS 162 CITATIONS

All content following this page was uploaded by Michael Eisenberg on 08 April 2015.

The user has requested enhancement of the downloaded file.
Computers and Physical Construction: 
Blending Fabrication into 
Computer Science Education

Michael Eisenberg, Ann Eisenberg, Leah Buechley, and Nwanua Elumeze
Dept. of Computer Science, University of Colorado, Boulder CO USA

Abstract
In this paper, we argue that physical fabrication can be productively interwoven into a computer science curriculum. We present two project-based courses that we have taught: Things that Think, a course in the creation of novel educational artifacts for children, and Leonardo da Vinci: the Engineer, a course devoted to the engineering work of the great Renaissance artist. Both courses involve a substantial degree of computer-aided physical fabrication. In the first course, projects focus on embedding computation into physical objects; in the second, the focus is on recreating great ideas of engineering through the use of up-to-date construction tools and devices. Toward the end of the paper, we use our two courses to ground a discussion of the role that material construction could eventually play in the broader computer science curriculum.

Keywords
Computer-controlled fabrication; computer science education, embedded computation.

1. INTRODUCTION: COMPUTERS AND PHYSICAL FABRICATION

In many universities—our own included—the department of computer science is housed within a larger school of engineering. At our own university, at least, the placement is not without a bit of tension. Whereas other engineering departments (chemical, mechanical, civil) actually make a practice of building—of creating tangible products—the computer scientists work in what others regard as the rather ephemeral medium of software. Indeed, Frederick Brooks, in his classic book The Mythical Man-Month, asserted that this very "intangible" side of computer science is what sets it apart as a discipline:

The programmer, like the poet, works only slightly removed from pure thought-stuff. He builds his castles in the air, from air, creating by exertion of the imagination... [1, p. 7]

For many topics within computer science, Brooks' observation is as true today as it was over thirty years ago. Nonetheless, programming need not always have ephemeral results, nor need it always be one step removed from the physical world. Increasingly, the advent of a wide variety of small-scale inexpensive fabrication tools has expanded the productive range of computer science. Computers can now be used to manipulate stuff—to create complex or novel artifacts that would have been prohibitively expensive or difficult a decade ago. [Cf. [3]] Through the medium of fabrication, computers enable a tightly interwoven mixture of software engineering and physical engineering. The programmer not only builds castles in the air: he or she may build real-life castles as well.

This paper describes two courses that we have taught here at the University of Colorado—one a computer science course, one a general engineering course—that blend the use of computers with physical construction. Both courses emphasize a view of computers as a central (but not isolated) tool in the creation of tangible objects; and both courses also introduce students to a range of powerful computationally-controlled fabrication devices. In effect, the computer is seen as the heart of a new type of "shop": a place in which programs and physical objects are created and combined to the mutual enrichment of each. Moreover, these courses suggest the possibility of still broader long-term shifts in the structure of computer science curricula.

The two courses will be discussed in far greater detail shortly, but it is worth introducing them briefly here. The first, "Things that Think", is a well-established one-semester computer science course aimed primarily at upper-level (junior and senior) engineering majors and graduate students. "Things that Think" (or TTT, as we have dubbed it) focuses on the creation of new types of educational artifacts that incorporate elements of embedded computation; students create mechanical toys and construction kits whose behavior or interaction is enhanced
by the use of microcontrollers, sensors, and actuators. The second course, "Leonardo da Vinci: the Engineer" is a recently-created one-semester course (taught for the second time in fall 2007) aimed primarily at first-year engineering students. In this course, students explore the history of engineering by using computer and fabrication devices to create working replicas of (often gorgeous) engineering devices taken from Leonardo's notebooks.

Between the two courses, there is a stylistic emphasis on merging computation and physical design. The curricular content of the two, however, is intriguingly different. TTT explores the design of children's artifacts from an educational standpoint: what makes for a tasteful, entertaining educational toy? The Leonardo course focuses on the history of engineering and includes lecture materials on da Vinci's life and the role of engineering in Renaissance society more broadly. For the purposes of this paper, the two courses illustrate that the stylistic decision to blend computation and physical construction lends itself in fact to a wide range of course topics and curricula—a point to which we will return toward the end of this paper, where we discuss how to weave the "shop mentality" into a larger view of computer science education.

The remainder of this paper will be organized as follows: the next two sections will focus on the two courses and discuss the role that fabrication plays in each. The fourth and final section builds upon these earlier descriptions and explores the role that fabrication may eventually come to play in computer science and engineering education.

2. THINGS THAT THINK:
CREATING COMPUTATIONALLY-ENHANCED ARTIFACTS FOR CHILDREN

Things that Think is a course that originated almost a decade ago as of this writing (we are offering the course for the ninth time in the current spring 2008 semester). The focus of TTT is the construction of novel, computationally-augmented artifacts for children, primarily with an eye toward mathematics and science education: mechanical automata, scientific exhibits and displays, construction sets, and so forth. Over its history, the course has evolved to take greater advantage of available fabrication devices; in its earliest versions, students made heavy use of traditional shop tools for construction. In the current incarnation of TTT, by contrast, working with and learning about computer-controlled fabrication devices constitutes a major element of the course curriculum.

The major assignments in TTT consist of two projects (students typically work in small teams of 3-4 members). The first of these projects is to create a mechanical automaton; by way of illustrations, we introduce students to historical examples by such creators as Jacques Vaucanson [4], and also show current work by (among others) the artists associated with the Cabaret Mechanical Theatre. [2] The students' job is to create their own working automaton, in whatever artistic style they prefer, with the single constraint that it must incorporate some significant use of embedded computation in its design. This latter constraint, again, may take a variety of possible forms: the automaton might make use of computer-controlled sensors or motors as part of its operation; it
might incorporate a degree of randomness in its operation, as dictated by a computer program; it might offer its viewer (or "user") some degree of interactivity; it might employ multiple computational elements that communicate wirelessly; and so forth. We introduce the students to the "cricket" devices created by the MIT Media Lab [6] as a default means of incorporating computation into their projects; but if they prefer, students can use other computational platforms such as PIC microprocessors or the recent, popular Arduino platform.

In recent years, the second TTT project has involved the creation of a computationally-enriched construction toy. Again, very wide latitude is given to the students regarding what may be regarded as "construction"; a variety of historical examples are presented (including obvious choices such as Lego and Meccano as well as more domain-specific sets such as chemical models and architectural building blocks). And as before, students are offered a wide range of strategies for incorporating computation into their designs. For example, pieces in a kit might communicate with each other to produce interesting effects; or they might communicate with desktop computers; or alternatively, the user might be given an interface with which to print out a variety of customized construction pieces.

As they build their projects, students in TTT are introduced to several means of computer-aided fabrication. Our lab has two 3D prototyping machines—one that outputs solid shapes in plaster, and the other in plastic. In either case, the student may use commercial software (such as Rhino) to design three-dimensional objects; we also instruct the students in the STL format in the event that they wish to write programs to create their own three-dimensional forms. The lab has a desktop laser cutter as well; this may be used to cut shapes from wood, acrylic, cardstock, or fabric. Students specify shapes directly in Corel Draw, and again we show them how to program their own shape specifications in HPGL format. Our lab also has a computer-controlled sewing machine (for creating custom embroidery patterns), two-dimensional milling machine, color printer, and a variety of desktop shop tools including a scroll saw, drill press, and sander. Finally, the lab has a chemical hood in which students may make safe use of solvents and adhesives.

Figure 2. A portion of a "wine-glass orchestra" automaton. In fact, this automaton consisted of three separate noise-making constructions, each activated by a cricket-controlled motor. When the wine-glasses in this portion of the automaton are about to touch, the "wine-glass" cricket sends out an infrared signal turning off the other two portions of the construction and ending the performance.

For many of the computer science students (including graduate students) who enroll in TTT, the course represents the first time that they have been challenged to combine computation with physical materials; indeed, this is often their first tangible construction experience of any sort. Nonetheless, we have been delighted with both the enthusiasm and engineering creativity of students who heretofore worked primarily in the medium of Brooks' "thought-stuff". Figure 1 shows views of a "golf-playing" automaton: a photo of the overall construction, and an interior view that depicts the cricket controller inside the device as well as a worm gear mechanism. When activated, the golfer figure would tap a metal golf ball off a tee toward a hole; the ball would drop into the hole and return to the tee via a motor-driven ramp. The device employed a light sensor and magnetic sensor to detect the placement of
the ball; when placed on the tee, the ball activated the magnetic sensor, and the computer controller would cause the golfer to swing his club. When the ball reached the bottom of the ramp, it blocked a light sensor, and the computer would move the ball upward on the ramp back to the tee. Figure 2 depicts one portion of a remarkable "wine-glass orchestra" construction that made whimsical sounds using utensils and glasses. This construction made use of multiple cricket controllers that communicated via infrared signals; each controller operated a motor for its own noise-making portion of the device. When the "wine-glass toasting" portion (shown in the figure) was about to touch its glasses together, it would signal the other portions to cease their operation. Figure 3 shows a less elaborate (but still lovely) cricket-controlled automaton from this semester's version of the course: a "growing flowers" device that, when stimulated by a light sensor, causes glow-in-the-dark flowers to "bloom" mechanically.

Figure 4 shows two photographs from a recent example of a construction-kit project in the course (i.e., the second of the two course projects). This is a construction set for botanical forms based on the mathematical formalism of L-systems. [5] Here, the construction kit itself contains no computational element, but the user is able to print out "snapper pieces" (shown at the top) and branches in wood on the laser cutter. Although a relatively simple prototype, the project illustrates how construction kits may eventually be augmented or customized through personal fabrication devices: users can print out new, unusual, or additional construction pieces on their own.

![Figure 4](image1.png)

**Figure 4.** Two views of an "L-system construction kit". At top, a collection of laser-cut "snapper" connector pieces in wood; at bottom, a laser-cut tree construction of connectors and branches.

![Figure 5](image2.png)

**Figure 5.** A crankshaft operated by a worm gear based on a sketch by Leonardo da Vinci. The large blue acrylic gear was cut with a laser cutter; the white helical portion of the worm gear was produced with a three-dimensional prototyper.

### 3. LEONARDO DA VINCI: TEACHING THE GREAT IDEAS OF ENGINEERING THROUGH PHYSICAL DESIGN AND CONSTRUCTION

The second course, "Leonardo da Vinci: the Engineer", introduces incoming engineering students to important historical ideas in engineering through the use of design and computer-aided fabrication. Students explore the notebooks of Leonardo da Vinci, and we discuss the problems that Renaissance engineers had to face and the tasks that they might be called upon to undertake. In many cases, Leonardo's beautiful sketches were never realized in physical form (in at least some cases, there is continuing debate over the feasibility of his designs); the students' job
is to create working prototypes of his designs, making use of the fabrication tools and devices in our lab. Here, the computational aspect of the course consists primarily of instruction in computer-controlled fabrication; unlike the TTT course, the actual constructions (based, as they are, on Renaissance ideas) do not incorporate embedded computational control. At the same time, the course is intended to interest engineering students (particularly those who are undecided as to their eventual major) in the ways in which computer science can be a central, foundational discipline with which to begin a career in engineering.

Figures 5 and 6 show two constructions, both created by the same team of students, in the course. The first is a model of a worm-gear driven crankshaft closely modeled on a sketch of da Vinci's (although the materials give the construction a decidedly modern look). The second is a working model of a paddleboat based on a sketch of one of da Vinci's contemporaries. In both these constructions, students made use of both the 3D prototyping machine to print out elements in plastic, as well as the laser cutter to produce pieces in plastic and wood.

Figure 7 depicts yet another project from last fall's version of the course: a pile driver (optimistically equipped with a pet mouse to supply power). The structural pieces of this Leonardo-inspired, but modern-looking device were designed and cut in clear acrylic on the laser cutter. This and other projects from the more recent version of the course may be viewed in video form on the Web (see the references section).

4. WEAVING THE PHYSICAL WORLD INTO COMPUTER SCIENCE EDUCATION

Having described our two courses, we can now discuss the role—in our view, an increasingly important and fascinating role—that physical fabrication can or should play in computer science education. Certainly, our own computer science students often experience both a certain degree of stressful unfamiliarity and (on happier occasions) delight at dividing their creative energies between the screen and the workbench. Courses of this type are, in a sense, emotionally empowering (when they work): that is, they give computer scientists a degree of confidence in their own breadth of skills as overall engineers. We believe that this is in fact representative of a type of construction that will increasingly find its way into home workshops, garages, school laboratories, and other settings: namely, a style of work that seamlessly blends programming, fabrication, and traditional shop work.
At the same time—because this type of construction tends to fall between disciplinary boundaries—a skeptic might argue (with some reason) that these sorts of laboratory courses should be introduced outside of computer science. For instance—to pursue this argument—a mechanical engineering course might be responsible for introducing 3D printing to produce mechanical designs. Or perhaps, a designated robotics course (whether in computer science or mechanical engineering) should be the venue for these techniques.

We welcome the growing prevalence of robotics courses and fabrication projects in other departments; but at the same time, we argue that physical construction has (or could well have) a much more productive and prominent role in the "mainstream" computer science curriculum, to the mutual benefit of computer scientists and fabrication enthusiasts. That is, computer science has something to offer the study of fabrication in engineering, and conversely fabrication has something to offer the computer science curriculum.

To expand on that first claim: introducing computer science students to fabrication tools allows for a variety and depth of projects that would be unusual in other disciplines. In our TTT class, students have often written surprisingly complex or sophisticated programs to control their devices—programs that would likely be beyond the skills of students in other disciplines. On occasion, TTT students have created artifacts that combine fairly large-scale original desktop programs with smaller-scale microcontroller programs—again, something unlikely to occur in other disciplines. By engaging computer science students (with advanced programming skills) in the design of physical artifacts, the TTT course serves as a venue for brainstorming projects that combine tangibility with potentially complex patterns of response. Moreover, we have noticed another unexpected effect of teaching fabrication techniques to computer science students: often, those students bring to the study of fabrication a critical eye for software design. As a result, we sometimes find ourselves discussing and critiquing the design and interface of the fabrication software itself.

At the same time, to pursue the second claim, construction projects offer an important perspective to the student of computer science. Certainly, writing a program to control a physical device highlights certain types of programming issues that are often suppressed elsewhere. For instance, when a real motor is told to turn a shaft for 60 degrees, the performance of the actual device is never quite as exact as a screen simulation might suggest; the programmer has to allow for numerical inexactness in virtually every aspect of a working program. When computation is blended with construction, the student often has a chance to see how one side of the task can make the other side easier or more difficult: a better or more elegant physical design can allow for a simpler software strategy. Arguably, these are lessons learned in robotics classes as well; but again, while we are pleased to see robotics classes in the computer science curriculum, the range of physical construction now available to computer science courses goes well beyond that usually associated with robotic design. Computer scientists can work in fabric, paper, and plaster—unusual materials for a robotics project—and can find themselves concerned with aesthetics in a way that does not generally occur in robotics courses. The range of constructions designed in the TTT and Leonardo courses illustrate how varied, how stylistically fluid, the new fabrication techniques can be.

Over time, we believe that physical construction will find its place in a larger portion of the overall computer science curriculum. For example, a wide variety of novel "smart" materials—shape memory alloys, electroluminescent wires and films, electrorheological fluids— dovetail well with computational control. (In fact, one current project in this semester’s version of TTT makes prominent use of "ferrofluid", a colloidal mixture of magnetic particles in oil, to produce startling visual effects under computer control.) We could imagine, going further, a course that introduces exciting ideas in materials science to computer scientists in the context of writing programs for novel types of "stuff". Likewise, one might imagine a greater emphasis in human-computer interaction upon ambient computational interfaces (in contrast to desktop interfaces). Computer graphics effects might be realized in media other than a flat screen (e.g., displays may be placed or distributed on the surface of a customized physical construction).

Computer programming, undeniably, will always have a strong element of the ephemeral, as Brooks observed decades ago. At the same time, there is an increasing range of personal and engineering styles open to the professional computer scientist: while some might concentrate on desktop software, others might concentrate on "physicalized" software tightly interwoven with real-world construction. As this range of professional styles evolves and expands, it will (we believe) come to be reflected in computer science curricula as well, affording students the opportunity to build both in "thought-stuff" and in "justplain-stuff" to suit their own creative impulses.

5. ACKNOWLEDGMENTS

The golfer automaton was created by Curtis Caravone, Erin Carroll, and Phil Dressen; the wine-glass orchestra by Leah Buechley (as a student in the class); the flower automaton by Dan Maples, Christina Wolfskill, and Matthew Hilgers; the L-system kit by Ben Farkas, Nick Romanyszyn, and Patrick Clary; the first two da Vinci models by Jaime Catchen, Andrew Gordon, and Ben Gunden; and the pile driver by Mark Baker, Nathan Madigan, and Chris Scott. Mark Gross collaborated in originally creating the Things that Think course. The work in this paper was partially supported by the National Science Foundation under grants ITR-0326054 and REC-0125363.
6. REFERENCES


Websites for the two courses may be reached from the following site:
13d.cs.colorado.edu/~ctg/Courses.html

Videos of representative recent projects in the Leonardo da Vinci course may be viewed at the following websites:
http://www.youtube.com/watch?v=T3heJuY5Is
http://www.youtube.com/watch?v=L_InxRLGiY
http://www.youtube.com/watch?v=ByCjXAmsJUsC