Plushbot: an Application for the Design of Programmable, Interactive Stuffed Toys

Yingdan Huang and Michael Eisenberg
Department of Computer Science
University of Colorado, Boulder CO USA
Yingdan.Huang@Colorado.EDU

ABSTRACT
In recent years, a burgeoning landscape of tangible, interactive toys has emerged–construction kits, robotic characters, and so on. At the same time, there has been relatively little development in software systems that permit users (including younger users) to design and create their own computationally-enhanced tangible figures. This paper presents Plushbot, a prototype system whose purpose is to allow people to create and customize programmable stuffed toys. Plushbot includes features specifically aimed at facilitating the incorporation of computational elements within stuffed fabric designs. Here we describe the current (still early) state of the system, show a completed project employing the system, and outline near- and medium-term directions for its continued development.

Author Keywords
Tangible interfaces, stuffed toys, Plushbot.

ACM Classification Keywords
H5.m. Information interfaces and presentation (e.g., HCI): Miscellaneous.

INTRODUCTION
The area of toy design has undoubtedly been a fertile one for research and development in tangible computer interfaces. Recent years have seen the development of all sorts of computationally-enhanced toys: responsive robotic figures such as Furby [cf. 5], powerful computational construction kits [cf. 7] such as Topobo [6] and Posey [8], educational devices such as "talking globes", and many more. In general, such artifacts have vastly expanded the range of interaction that youngsters are capable of having with tangible play materials.

At the same time, however, there has been relatively little software development allowing users (including young users) to create their own personalized, computationally-enhanced toys. In general, such toys tend to be pre-programmed (such as Furby); or, alternatively, when the child does the "programming" (as with construction kits) she employs the pre-designed pieces and materials supplied to her. Undoubtedly these models of play are pleasurable and creative; but the toys themselves are not objects of the child's creation.

This paper introduces a software system, Plushbot, whose purpose is to allow people to create their own interactive stuffed toys. In essence, Plushbot is a design application through which users can interactively place computational elements on the construction template for a plush toy; the system then assists the user in the actual physical creation of the toy itself. Plushbot is still, at present, in a very early stage; and there are numerous enhancements and likely directions for future development of the system, even in the near term. Nonetheless, the system—even in its current rather embryonic state—raises a variety of interesting questions for the larger project of permitting users to design and create their own interactive, programmable computational artifacts.

The remainder of this paper is organized as follows: in the second section, we describe the current Plushbot interface and show a working stuffed toy created with the system. The third section of the paper uses the current implementation of Plushbot as a springboard for a wider-ranging discussion of issues related to user-initiated design of computational toys. This final section also discusses some relevant related work, and sketches some of the planned future directions for enhancing and extending our current system.
Figure 1. The Plushbot screen interface, in the context of a construction scenario. The main portion of the screen depicts the pieces of a sample construction (outlined in blue), extended by LilyPad Arduino components (selected from the menu of pieces at right), and linked by purple lines that denote the placement of conductive threads. The various features of the interface are described in the accompanying text.

PLUSHBOT: A DESCRIPTION AND SCENARIO

The easiest way to introduce Plushbot is through a construction scenario, as illustrated by Figure 1. Here, the user has begun by choosing a pattern for a plush toy to create (in this case, a cat). The various pieces for the toy are depicted on the screen in outline form; these outlines represent the fabric cutouts that will eventually be sewn together to create the final toy.

With the pieces visible on the screen, the user now selects icons for computational elements—processors, batteries, sensors, and actuators—and positions these on the outlines in drag-and-drop fashion. The computational icons themselves represent components from the popular LilyPad Arduino system [3], and are specifically designed for incorporation within textile-based craft projects. As positioned, the icons will be used to indicate where LilyPad components will eventually be sewn onto the fabric pieces. Briefly, for those unfamiliar with the LilyPad Arduino system: the components allow the eventual constructed plush toy to respond in programmed fashion to sensor inputs (e.g., ambient light levels, or a "squeeze" from the user at a particular spot) and to represent that response through a variety of actuators (e.g., flashing lights, speakers, motors). (See [2] for a discussion of e-textiles in the context of children's crafts.)

In short, then, the Plushbot user is not merely constructing a static toy, but is also constructing the design template for a potentially limitless range of distinct programmable artifacts. It should also be mentioned—as an aside—that to date, the vast majority of LilyPad projects focus on the creation of programmable wearables; as such, the use of the LilyPad within plush toys represents a (mild) stretch for the device and gives rise to several design issues to which we will return in the final section.

To continue with the scenario represented in Figure 1: the user now links the various LilyPad components using...
flexible line- and curve-drawing tools indicated at the top of the figure. The lines drawn in this case indicate to the user precisely where to sew conductive threads into the eventual fabric pieces; these threads effectively serve as the "electronic wire" for the computational elements of the construction.

Once the overall design of the plush toy is completed on the screen to the user's satisfaction, she now outputs the design of the toy in a format that can be directly sent to a laser cutter (this is indicated by the "output HPGL" icon shown in the figure). The output file created by Plushbot, once sent to a laser cutter, will cut out the various pieces in felt and etch (or engrave) markings on those pieces indicating where the LilyPad components should be placed, and where the conductive thread linking those components should be sewn.

Before proceeding any further, it should be noted that—simple as these capabilities are—the use of Plushbot to allow users to position computational components and their connections on fabric pieces on the screen represents a valuable type of assistance for LilyPad users, as any practitioner knows. In general, the most difficult step for the creation of an e-textile artifact (such as a LilyPad-enhanced garment) is in visualizing where components will be placed and how they will be connected by sewn threads. Not infrequently, the placement of conductive threads gives rise to errors in the operation of the eventual construction (inelegant paths, short circuits, and so forth). Thus, the Plushbot interface provides a valuable template for planning out e-textile design in a way that reflects actual practice among designers.

Once the felt pieces have been cut out and etched by the laser cutter, they constitute a customized template for sewing on the LilyPad components and conductive thread connections. The pieces are then sewn together at the boundaries (as in standard plush toy design) and filled with padding; the LilyPad program is downloaded to the sewn-on processor; and the result is an interactive plush toy. In the scenario shown here, the stuffed figure permits the user to light up the eyes by squeezing the toy in its "belly".

Figure 2. The finished plush toy created from the template shown in Figure 1. A touch sensor in the middle of the toy allows the user to give the character a gentle squeeze that causes the LEDs in the eyes to light up. At right, a back view of the toy shows the sewn-on microprocessor (at top) and battery component. Note also the conductive threads that are sewn into the processor and that lead to the various other computational/electronic elements of the toy. A video of the working toy can be found at: http://www.vimeo.com/13783319

REFLECTIONS ON PLUSHBOT: RELATED WORK AND POTENTIAL FUTURE DIRECTIONS

Plushbot is, as noted, a relatively early working prototype at this stage, and is not ready for widespread release; but even now, it raises a variety of interesting questions worth exploring.

First, it should be noted that in its current version, Plushbot does not include features for designing novel cutout forms
for plush toys, in the manner of the recent (and beautiful) "Plushie" software system. [4] That is, Plushbot—at present—makes use of pre-existing fabric cutout templates (in the near future, our system will include a small library of such templates). Combining the features of Plushie with those of Plushbot would, arguably, result in a whole much greater than the sum of its parts: a system that could be used to create wholly original (in form), programmable, interactive toys.

Yet another likely addition—or augmentation—to Plushbot is a programming and simulation capacity. That is, the Plushbot user does not merely wish to place icons of LilyPad components on the cutout forms shown on the screen, but to write programs and simulate them; this would permit the user to envision how the eventual plush toy will look and behave before she goes to the trouble of sewing the object together. We plan to incorporate some programming/simulation component into Plushbot within the near future; very likely this will involve combining our application with a LilyPad simulation system currently under development elsewhere (as opposed to creating a complete system of this sort ourselves). In other words, Plushbot will be designed to combine with other applications rather than as an attempt at a "universal" standalone application.

The previous two paragraphs are suggestive of a larger important theme that is worth at least mentioning here, given the available space: namely, that Plushbot is deliberately designed with an eye toward its placement in a larger technological "ecosystem"—a system of possibly supportive design tools (such as Plushie), language systems, hardware components (the LilyPad), and fabrication devices (the laser cutter). This design philosophy is in contrast to a style of "standalone" software design that attempts to incorporate all functions and features within itself, and that assumes a minimum of external supportive hardware. Again, this is a principled decision, and reflects a view of multiple technologies—both software and hardware—as constituting a mutually supportive web of functions. This view is substantially inspired by the eloquent discussion of technological advancement in Arthur [1]. In this view, "pieces" of already-existing technology can be reconfigured, recomposed, and extended to produce new functionality. Plushbot thus exists within—indeed, only makes sense within—a larger surrounding infrastructure of devices and tools.

At the same time, the Plushbot system—by focusing on the design of interactive plush toys in particular—raises design issues that are specific to its domain. Just to take a simple example: consider the sample toy shown in Figure 2. Here, the computational elements (prominently including processor and battery) have been placed on the exterior of the toy. In certain projects it might, of course, be desirable to place these components on the inside of the toy, within the padded region (e.g., to avoid incorporating these elements into the toy's character, or simply for aesthetic reasons). In such a case, however, it would still be advisable to allow the user to access the computational elements—say, to change a battery or reprogram the LilyPad. These considerations result, in turn, in slightly different constraints on plush toy design: namely, we might wish to create a toy that can be easily "opened up" for access to its interior (an atypical consideration for this type of toy, and one not usually encountered in the "wearables" projects that characterize most LilyPad tutorials).

It should thus be clear that the design of a tool such as Plushbot—a tool that permits users to create an identifiable (but still quite varied and rich) genre of tangible artifacts—raises a host of still relatively unexplored questions. These questions will be the focus of our near-term work on the system; and once the system is released for more widespread use, it will be possible to study and assess the way in which Plushbot is employed in combination with a continually burgeoning technological landscape.

ACKNOWLEDGMENTS
The work described in this paper has been funded in part by the National Science Foundation under grant CNS0940484. We wish to thank Leah Buechley, Mark Gross, and Nwanua Elumeze for helpful conversations.

REFERENCES
3. LilyPad Arduino:
http://www.arduino.cc/en/Main/ArduinoBoardLilyPad