PHYSICAL COMPUTING AND MULTIMODAL INPUT IN HUMAN-COMPUTER INTERFACES*

Lori Scarlatos
Department of Computer and Information Science
Brooklyn College
2900 Bedford Ave.
Brooklyn, NY 11210
Tel: (718) 677-6170
Email: lori@sci.brooklyn.cuny.edu

Tony Scarlatos
Department of Computer Science
Stony Brook University
Stony Brook NY 11710
Tel: (631) 632-8432
Email: tony@cs.sunysb.edu

ABSTRACT

The human-computer interface is widely recognized as an important part of any software project. Consequently, the principles of human-computer interaction are increasingly being taught in computer science departments. Usually, this type of course will focus on the design, implementation and testing of graphical user interfaces. Yet today's ubiquitous computing applications require interfaces that are context aware and respond to more than text from a keyboard or the click of a mouse. Students who have experience developing interfaces with a range of multimodal inputs will be better prepared to meet this growing need.

This paper describes a new course that is being taught at two different colleges at the undergraduate level. Students in these classes learn the theory behind good interface design while obtaining practical experience implementing applications that respond to speech, sensor input, and computer vision. This paper also describes how students go on to apply this knowledge to educational applications in subsequent independent projects, working with teachers as clients. Finally, the paper presents several of the projects produced by students in these classes and in subsequent collaborations.

INTRODUCTION

Recent innovations in computer-human interfaces have greatly expanded the realm of possible applications for computers. With these interfaces, computers can respond to one's natural actions within an environment, such as building with blocks, playing with 3D puzzles, or dancing on a stage. These innovations also make it possible for more than one person to interact with a computer at a time, paving the way for more collaborative group activities. Furthermore, these alternative approaches to computer-human interaction can make computers more accessible to people who may have difficulty with more traditional interfaces.

Educational applications are wonderful candidates for these innovative interfaces. Studies of how people learn [1, 3] indicate that different people benefit from different strategies for developing understanding of a subject. For example, some students gain a deeper understanding of a subject when they engage in group discussions or activities, Others find lessons more meaningful when they are engaged in physical activities. In particular, math teachers now commonly use math manipulatives – props and puzzles that teach mathematical concepts – to help their students to understand the principles and standards being taught [9]. By presenting information through different media, and providing students with an opportunity to work together on physical activities, we can greatly increase the changes that students will actually learn what they are supposed to.

Researchers in the area of physical computing have developed several applications that make use of this type of technology to enhance learning. Programmable cubes, beads, and badges [5], which allow children to experiment with communications systems, led to the development of Lego Mindstorms. Other electronic building blocks allow children to build 3D models that automatically generate 3D computer models [4]. Tangible interfaces for collaborative learning environments [6] provides hints and help as children use physical Tangram pieces to solve geometry problems. Curlybot teaches children to think sequentially by remembering a sequence of moves that a child puts it through [2]. I-sign helps hearing impaired children to learn to read by enabling their parents to read out loud to them [7].

Our goal has been to build on these recent innovations and apply them to an area of great national importance: math and science education. While computer scientists and engineers have been advancing the technology, educators have been pioneering the use of manipulatives and collaboration as learning tools. Our objective is to bring together the best of both worlds, in research and education that is as relevant as it is novel.

THE COMPUTER SCIENCE COURSE

Although many computer science programs offer courses in computer-human interaction, most of these courses focus on interfaces that use a mouse or keyboard to manipulate on-screen objects in a windowing environment. Yet by forcing people to use computers solely in this fashion, we are severely limiting the range of possible applications. Computing is becoming more and more ubiquitous: we are finding it in our automobiles, our television sets, and even our homes.

We have developed an upper-level undergraduate course that focuses on alternative computer-human interfaces that use state-of-the-art technologies. Our hope is that by exposing computer science students to the range of possibilities, we will enable them to contribute significantly to the development of new interface paradigms where the user may not even be aware that a computer interface is there.

Theoretical

Half of the class time is devoted to the study of concepts, theories, and techniques employed in the development of human-computer interfaces. This provides material for class discussions, exams, and a term paper.

A large chunk of this time is spent on material that is appears in traditional human-computer interaction curricula [8]. We discuss the range of paradigms used in human-computer dialogs – including command languages, menu interfaces, and direct manipulation – focusing on the benefits and drawbacks of each. In our discussions of interface design methodologies, we present the usability engineering approach which includes user/task analysis, scenario development, requirements analysis, storyboarding, rapid prototyping, and iterative development. In a section on multimedia representations of information, we present examples and guidelines for the effective use of graphics, animation, audio and text in computer-human interfaces. We talk about effective feedback and dialog design. Finally, we discuss different approaches to usability testing, including analytic versus empirical methods of evaluation, and test design and analysis.

The rest of this time is spent on topics related to physical and ubiquitous computing. These include tangible user interfaces, virtual and augmented reality, and computer supported collaborative work. However, we focus most of our time on three different sources of input: speech, computer vision, and sensors. In each of these areas, we provide examples of how they can be used in an interface. For example, in speech, we compare continuous natural speech recognition to discrete recognition of limited vocabularies. In vision, we talk about gestural interfaces, recognition and tracking of objects, and motion detection. With sensors, we talk about the various types of sensors (including touch, bend, light, audio, and magnetic) and the types of inputs that can be derived from them.

We also present the fundamental technologies and algorithms used in each. Although we do not have the time to go into the algorithms in great depth, the material presented is sufficient to give our students an understanding of the different approaches that can be taken. More importantly, they learn why one approach would be used over another, based on the conditions of the environment, requirements of the application, and expectations of the users.

During the semester, students are asked to write an essay on how they imagine computing will be used that their university ten years from now. This includes computing devices in the classroom, the dorm rooms, and on their person, used for work, study, communication and entertainment. Although students are encouraged to use their imaginations, we ask them to back up their predictions with evidence (from books, journal articles, newspaper stories, et cetera) that these predictions will likely be realized.

Practical

The other half of the class gives the students practical experience designing and implementing applications that respond to a variety of inputs. We chose to have our students develop these applications using Macromedia Director on the Macintosh platform. Director is a multimedia authoring tool that is widely used in industry for rapid prototyping. A drag-and-drop interface allows users to quickly put together a user interface that incorporates a wide variety of multimedia, including graphics, audio, animation, and video. Director also provides a powerful programming language, called Lingo, that allows developers to specify how the program should respond to events. In addition, Lingo is easily supplemented with third-party software libraries known as Xtras, which provide additional functionality. In our case, these Xtras allow us to access – and respond to – a much wider range of inputs.

Speech recognition is supported on all Macintosh computers through the operating system's "speakable items" capability. Using AppleScript, the Macintosh OS scripting language, applications can respond to a limited set of words. Yet although the number of words is limited (to about 20 per application), the range of words is not: one student group has developed an application that recognizes Spanish and Italian words, which have been spelled phonetically. The advantage of this speech recognition software is that it does not require any training, and therefore our applications can be used by any user. In working with this capability, our students learn the limitations as well as the possibilities of speech recognition. For example, they learn that polysyllabic words are easier to recognize than monosyllabic words. Our students are able to integrate speech recognition with their Director applications by using the zScript xtra.

We incorporate computer vision into Director applications using the TrackThemColors xtra. This xtra provides functions that process images from an ordinary webcam (using USB) or a digitial video camera (using Firewire). These functions support finding spots of a given color, finding the brightest spot, tracking points of a given color, and detecting motion in an image region. A tolerance is given in most cases, allowing for

Figure 1. Floor mat as number line

Figure 2. Glove with sensors

varying levels of precision in the results. This is very important, as changing light levels in a classroom (from, for example, the sun going behind a cloud) can drastically alter the color values of the pixels being detected. This also provides an important lesson on the ins and outs of using computer vision in an interface.

Sensors provide input to Director applications in two different ways. First, we built a set of floor mats with binary touch sensors that are connected to a keyboard emulator. Thus, stepping on a particular spot on one of these mats is comparable to typing that key on the keyboard. One of these mats has the sensors lined up in a single row. This can be used as a number line, keyboard, or placeholders in a binary number. Another mat has sensors laid out in a 4 x 6 grid. Figure 1 shows a child on the number line. Second, we use a set of i-Cube sensors that plug into a midi controller, which in turn is connected to the computer via USB. For this option we have a potentiometer, bend strips, touch pads, and motion sensors. Figure 2 shows a glove that incorporates bend and touch sensors, which was used in an application that detects finger spelling.

At the two colleges, we have tried two different approaches in our programming assignments. At one, students are required to work individually on four different programs: one demonstrating a traditional graphical user interface, the second using sensor inputs, the third utilizing information from a video camera, and the fourth responding to speech inputs. Although students are allowed to combine all of these capabilities in one application, most students choose not to. At the other college, students work in groups on a single large project, designed for a specific audience. Although the students learn how to implement interfaces using all of the approaches, they generally choose to incorporate just one or two. We have had varying levels of success with both approaches.

COLLABORATION WITH THE SCHOOL OF EDUCATION

Although computer science students are capable of developing educational applications with these innovative interfaces, they are not necessarily well equipped to determine how an educational activity should be presented to children. For this, we need to collaborate with people who specialize in the education of children: teachers.

We worked with professors who teach graduate education courses to produce modules focusing on how computers can enhance education using a variety of interface modes. These modules were then integrated into existing courses at both colleges. These future (and often current) teachers then worked groups to develop ideas for how technology could be used in the lessons they give their own students. We had computer science students from the previous class work on the resulting applications with these education students, producing applications that the teachers could bring to the classroom. These computer science students received credit for an independent programming project, which is required for graduation.

RESULTS

Several promising applications have come out of these classes. We describe only a few here.

In one of the collaborations with the teachers, one of our students developed an application that uses computer vision to watch as children play with a set of pattern blocks. These pattern blocks – including triangles, rhomboids, trapezoids and hexagons – are used to demonstrate geometric ideas. Another student project that uses computer vision is a Tower of Hanoi project, which watches and provides hints as students work on that puzzle.

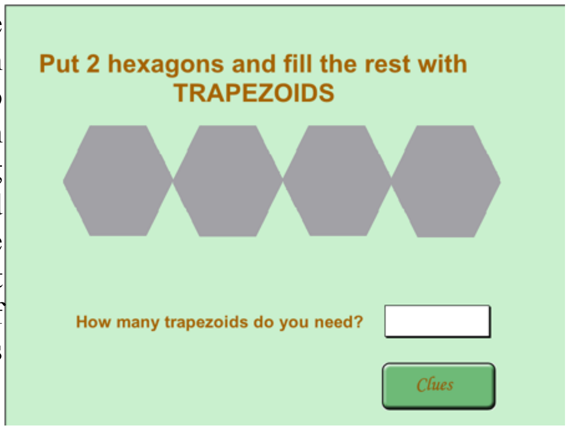

Figure 3. Geometry with pattern blocks

Figure 4 shows a screen shot from an application that helps students to practice their arithmetic skills. As math problems flash onto the screen, the children jump onto colorful squares on the floor (equipped with touch sensors) to answer the questions. A separate interface allows the teacher to customize the problems so that they are appropriate for the student using the application.

Figure 4. Math with Smart Step

Other applications using the floor pads taught fourth graders about the food chain, and about bird physiology. Our "number line" floor mat was also used as a keyboard, to teach people how to play a song. One application treated the row of sensors as a sequence of binary digits, asking students to create binary representations of decimal numbers by stepping on the appropriate places (toggling from 0 to 1 and back again).

FUTURE WORK

Two local museums, both specializing in math or science, have agreed to install some of the applications that our students have developed. In turn, we will be providing students (as paid interns) who act as "explainers" for the museums. This will give our students a unique opportunity to see their applications being used and evaluate the effectiveness of those applications.

ACKNOWLEDGEMENTS

We wish to thank our co-PI, Dave Ferguson, for all of his hard work and vision in bringing the educational component of this course to the Department of Technology and Society at Stony Brook University. At the same time, we wish to thank Eleanor Miele for all of her enthusiastic support, suggestions and insights; and for allowing us to invade her classroom and incorporate our materials into her curriculum. Finally, we wish to acknowledge the invaluable advice and support provided by members of our advisory committee: Preeti Gupta and Frank Signorello from the New York Hall of Science; Beth Deaner from the Goudreau Museum of Mathematics in Art and Science; Fred Gardiner, a mathematician from the CUNY Graduate Center; Michael Klopfer, an elementary school teacher and advisor to the Goudreau Museum; Rosamond Welchman from Brooklyn College; and Janice Grackin, our evaluator from Stony Brook University. This work is being supported by the National Science Foundation under a CRCD Grant No. 0203333.

REFERENCES

[1] J.D. Bransford, A.L. Brown, R.R. Cocking, M.S. Donovan, J.W. Pellegrino (editors), How People Learn: Brain, Mind, Experience, and School, National Academy Press, Washington, D.C., 2000.

[2] P. Frei, V. Su, B. Mikhak, H. Ishii, "curlybot: Designing a New Class of Computational Toys", *CHI'00 Proceedings*, 129-136, 2000.

[3] H. Gardner, Frames of Mind: The Theory of Multiple Intelligences, Basic Books, New York, NY, 1993.

[4] Y. Itoh, S. Akinobu, H. Ichida, R. Watanabe, Y. Kitamura, F. Kishino, "TSU.MI.KI: Stimulating Children's Creativity and Imagination with Interactive Blocks", Proceedings of Creating, Connecting and Collaborating through Computing (C5'04), 2004.

[5] M. Resnik, F. Martin, R. Berg, R. Borovoy, V. Colella, K. Kramer, B. Silverman, "Digital Manipulatives: New Toys to Think With", *Proceedings of CHI 1998*, 281-287.

[6] L.L. Scarlatos, "TICLE: Using Multimedia Multimodal Guidance to Enhance Learning", *Information Sciences 140* (2002), 85-103.

[7] T. Scarlatos, L. Scarlatos, and F. Gallarotti. "iSign: Making the Benefits of Reading Aloud Accessible to Families with Deaf Children". In *Proceedings of the Sixth IASTED International Conference on Computer Graphics and Imaging*. Honolulu, Hawaii, August 2003.

[8] B. Shneiderman, Designing the User Interface: Strategies for Effective Human-Computer Interaction, Addison-Wesley, 1998.

[9] J.A. Van de Walle, Elementary and Middle School Mathematics: Teaching Developmentally, Addison Wesley Longman, Inc., 2001.