Cord Input: An Intuitive, High-Accuracy, Multi-Degree-of-Freedom Input Method for Mobile Devices

Julia Schwarz, Chris Harrison, Scott Hudson, and Jennifer Mankoff
HCII, Carnegie Mellon
5000 Forbes Ave, Pittsburgh, PA 15213 USA
{julia.schwarz, chris.harrison, scott.hudson, jmankoff}@cs.cmu.edu

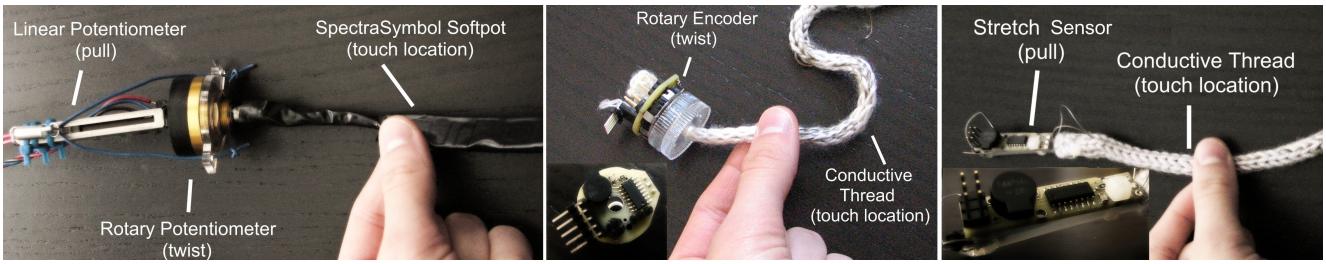

Figure 1. Iterative prototypes of a cord that senses how it is pulled, twisted, and where it is touched. Left: Version used for studies. Center: Second version with smaller rotary encoder, conductive thread, and integrated electronics (inset). Right: Third version with stretch potentiometer embedded in cord and integrated electronics (inset).

ABSTRACT
A cord, although simple in form, has many interesting physical affordances that make it powerful as an input device. Not only can a length of cord be grasped in different locations, but also pulled, twisted and bent—four distinct and expressive dimensions that could potentially act in concert. Such an input mechanism could be readily integrated into headphones, backpacks, and clothing. Once grasped in the hand, a cord can be used in an eyes-free manner to control mobile devices, which often feature small screens and cramped buttons. In this note, we describe a proof-of-concept cord-based sensor, which senses three of the four input dimensions we propose. In addition to a discussion of potential uses, we also present results from our preliminary user study. The latter sought to compare the targeting performance and selection accuracy of different cord-based input modalities. We conclude with brief set of design recommendations drawn upon results from our study.

Author Keywords
Mobile interaction, wearable computing, cord-based input.

ACM Classification Keywords
H.5.2 User Interface, B.4.2 Input/Output Devices .

General Terms
Human Factors, Design.

CHI 2010, April 10–15, 2010, Atlanta, Georgia, USA.

INTRODUCTION
Mobile devices with significant computational power are now easily carried with us. To access their functionality, people repeatedly pull these out from their pockets and bags to do something as trivial as change the volume of the currently playing song. Researchers have long explored how wearable computing can make such interactions more fluid. Commercial products, such as Apple's third generation iPod shuffle, now place buttons on headphone cords, hoping to reduce the time to reach for and manipulate the device. However, buttons provide a small number of discrete inputs, and in order to keep the devices on which they reside small, tend to be even smaller themselves.

In this note, we consider an alternative input method that is far more accessible and expressive than buttons. Rather than using small buttons specially integrated into headphone cords, one could appropriate the entire cord as the input device. Cords are particularly appropriate because they are often external and have a large surface area. Furthermore, while buttons provide primarily binary input, a cord could potentially provide continuous input along at least four dimensions (twist, touch location, and bend).

These four potentially continuous input dimensions a cord provides enable a wide range of applications. In addition to providing navigation and controls for audio players and other mobile devices, a cord could be used as a joystick to play eyes-free games on mobile devices. This cord could also be integrated into clothing to control devices or measure motion, and into everyday items such as backpacks and lampshades. Finally, an array of these cords could be woven into a cloth to detect how it is being manipulated.

To explore these interactions, we built a prototype cord that senses how hard it is being pulled, how much it is being twisted, and where it is touched. It is also possible to include bend as a fourth, independent dimension, but we leave this to future work. Our cord is not only far more expressive than a button; it is also easy to access, eyes-free, and unobtrusive.

RELATED WORK

There has been previous exploration into using cords and strings as input devices. Rantanen et al. uses a retractable string and squeezable display to enable menu navigation as part of a wearable interface [7]. Blasko et al. provides two dimensions (angle and pull distance) of input with a pull sensor and optical tracker [1]. Researchers have also built electromechanical systems using two rotary transducers (operating much like a joystick), and a retractable cord to measure pull distance [4, 5]. These allow the systems to localize the end of the string in free space (three dimensions).

The main commonality in this previous work is the focus on using retractable passive cords and sensors outfitted at one end. Our method, however, looks primarily at augmenting the cord itself (the cord is the sensor). Grossman et al. developed an input cord called ShapeTape which can capture two dimensions of potion: bend and twist [3]. Our approach allows us to capture three dimensions of motion – pull force, twist, and touch location. Unlike free space gestures, these can be combined in straightforward ways to provide highly accurate navigation and selection mechanisms. For example, twist to navigate a linear list, with pull (e.g., a quick tug) for selection. In fact, our technique could be combined with the aforementioned projects to provide additional expressiveness.

Evaluation of cord-driven interactions has been limited. To contribute to this area, we present results from our 14 participant user study, which evaluated three input modality combinations.

DEVICE PROTOTYPES

To explore our interaction concept we built a set of three prototypes. Our first prototype was fully functional, but did not have a final polished form (Figure 1, left), and was used for user tests.

We then constructed a pair of smaller prototypes which explored additional sensing approaches and solutions for size reduction. Our final prototype includes an integrated electronics package which could be placed inside a section of knitted cord. The first prototype used an Arduino (http://arduino.cc), a linear potentiometer and spring arrangement for sensing pull, a rotary potentiometer for twist, and a SpectraSymbol Softpot sensor for touch [8].

The remaining prototypes were implemented using a Cypress Semiconductor CY8C21234 PSoC microcontroller [2] on custom printed circuit boards. The cord was knitted using a hobby cord knitting machine. The cord knitting machine produced a hollow cord which could easily contain wires such as a headphone cord. In our implementation, very thin insulated wires are hidden inside it, connecting the sections of conductive thread (which form capacitive touch sensor pads) to the electronics. Twist sensing is done using a mechanical rotary encoder. In future work we hope to manufacture a small hollow rotary encoder.

Stretch sensing is performed using an elastic cord manufactured to increase in resistance as it is stretched [6]. We use a small section of this cord, protected by a bit of polyethylene tubing to prevent snags. This is in turn attached to a longer non-stretching string attached to the bottom of the stretchable knitted cord. In combination, these sensors make it possible to tell when the user is pulling on, twisting, or touching the cord.

EVALUATION

We conducted a user study to test which combination of twist, pull, and touch is most effective for targeting, and to test how the number of targets affects targeting performance in different conditions. Targeting refers to the combination of navigating to a target and selecting it. Performance includes speed and accuracy.

We recruited 14 participants (3 male, mean age 32 [SD=12.7]), who were compensated $10 for their time. After a brief overview of the project, participants sat in front of a table with the first prototype cord sensor hanging from one side. Participants were told use the cord sensor to navigate to a series of targets. Each goal target was presented to the user as a number on screen, along with information about how many total targets there were. After a short practice round, participants were told to navigate as quickly and accurately as possible to the goal. After navigating to one goal participants were immediately given another goal, and continued navigation from their previous target.

The sensor was located below the table and participants were instructed not to look at it, to simulate eyes-free use. Prerecorded verbal prompts were played to simulate a real-world audio menu. For example, if at target 2 and needing to go to target 5, twisting from 2 to 5 would cause the software to speak “3, 4, 5”. Once at a target, participants actuated the selection method for that condition (e.g., a pull or spacebar press). Targets were evenly spread along the virtual space represented by each continuous sensor, with no padding between targets.

As stated earlier, the prototype being tested could sense pull, twist, and touch, each on a continuous scale. The experiment was divided into six conditions, two for each type of sensing. In three conditions, selection was done with the spacebar so that the impact of sensing condition (*Twist*, *Pull*, or *Touch*) on navigation performance could be tested independently of selection. The other three conditions tested plausible combinations of navigation and selection all using the sensor, namely *Twist+Pull* (twist to navigate and pull to select), *Touch+Pull*, and *Pull+Twist.*

Each condition was divided into 4 blocks of 10 trials each (one block for each number of targets). *Touch* and

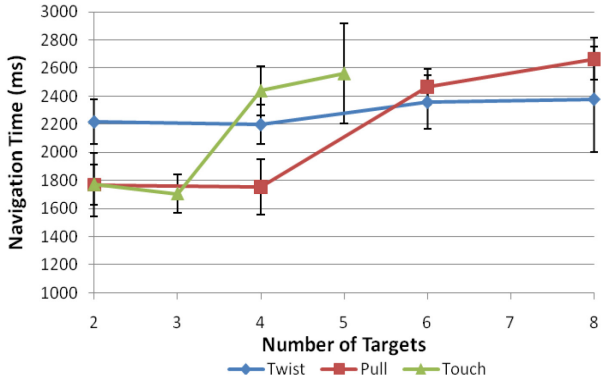

Figure 2. Navigation time for twist, pull, and touch conditions.

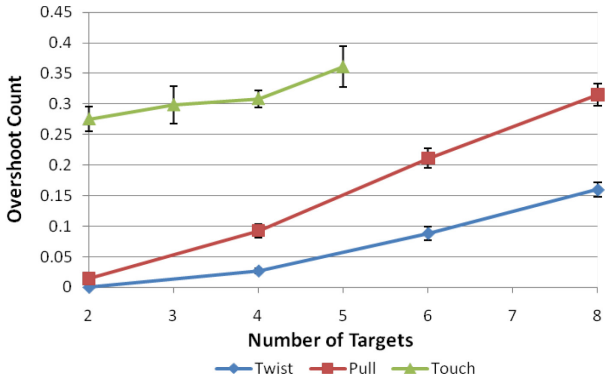

Figure 3. Overshoot count for twist, pull and touch conditions.

Touch+Pull, which we found were less sensitive during pilot studies, were subdivided into blocks containing 2, 3, 4, and 5 targets. All other conditions were subdivided into blocks containing 2, 4, 6, and 8 targets. Condition order was counterbalanced and block order was randomized. Within each block, the target to be selected for each trial was also randomized.

Measures: We calculated the time (ms) it took participants to complete a targeting task from trial start, when the goal target was presented on screen, to trial end, when selection was complete, averaged over each block (*targeting time*), the number of times a goal target was passed over before it was selected, averaged over each block (*overshoot*), and how many times participants successfully selected the correct target, calculated as a percentage *per* block (*success rate*). We chose to count overshoots because a large number of overshoots could indicate that the input technique was too sensitive. Because users were given audio feedback about their current target, we assume that they would all navigate to the correct target, and that failure to select a goal target is be due to the selection mechanism itself (i.e. accidentally twisting the cord as you pull to select). Our data provided insight into selection success rate, targeting error, and targeting time.

Data preparation: As just described, the measures were calculated at a per-block level. We checked for outliers by looking for data that was two standard deviations outside the mean per block on the targeting time measure. We found one outlier, which we attribute to a combination of hardware and user error. All data for the affected user were removed from our analysis.

RESULTS

Our results are broken into two parts. First we explore the impact of conditions (*Twist*, *Pull*, and *Touch*) on navigation performance. Then we explore which combination of sensors worked best for navigation and selection combined.

Navigation performance

In the *Twist*, *Pull*, and *Touch* conditions, the measure of targeting time is primarily a measure of navigation time, since selection is done with the keyboard space bar.

As expected, navigation time and overshoots increased as the number of targets increased (Figure 2, 3). *Twist* was most consistent (fastest, fewest overshoots) as the number of targets increased. Both the *Touch* and *Pull* conditions were about 50% slower when more than 3 (*Touch*) or 4 (*Pull*) targets were present. Surprisingly, *Touch* had much higher overshoot rates than pull and twist even for 2 targets, while the corresponding targeting times were mixed – being comparably as fast for 2 or 3 targets and slower for 4 or 5.

A lack of tactile feedback in the *Touch* condition made it difficult to identify different target locations. One participant in our study noted "It would be nice if we had some texture feedback about what area we were touching".

Performance of combined sensors

The average success rate across all users and blocks exceeded 93% in all conditions except *Pull+Twist*, which was only 70% [SD=8.5%] (Figure 5). We found this result consistent with user reactions. Several users complained that it was difficult to pull and twist at the same time. One participant noted, "The springiness [of the pull sensor] makes it hard to twist." After our study we piloted a small test to see if twisting had the same effect on touch location, and found that it did not seem to. Therefore, we believe that this low performance was due to the physical challenge of maintaining tension in a cord while twisting.

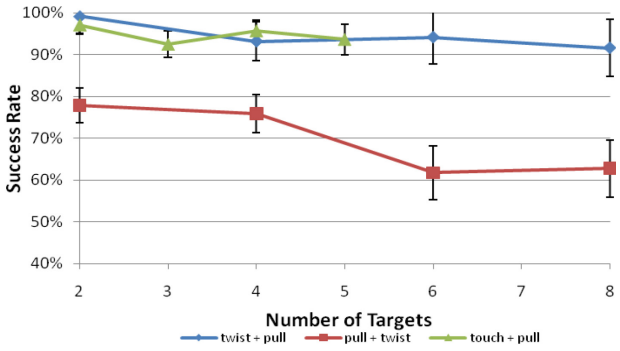

Figure 4. Success rates across conditions not utilizing the spacebar.

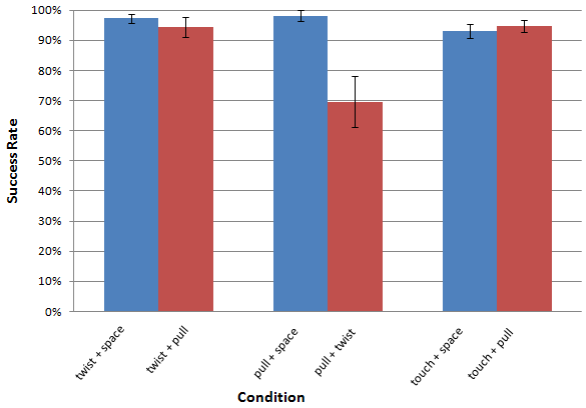

Figure 5. Average success rate across number of targets.

DESIGN RECOMMENDATIONS

We conducted a small survey with the same participants to determine which dimensions (*Twist, Pull, Touch*) worked best for continuous input (increasing the volume of a song) and for toggling values. We asked users to describe how they would use the cord to do one of two tasks, and then coded their descriptions as requiring twist, pull, touch, or several of these. Table 1 illustrates our results, and indicates that users thought twist was most appropriate for continuous input, while pull was most appropriate for toggling between values or making selections.

Based on results from our survey and user study, we would recommend that future implementations use touch location or twist for continuous input, and pull to toggle between values or make selections.

Task Name	Task Type	Twist	Pull	Touch
Increase Volume	Continuous	**60%**	13%	27%
Pause/ Play	Toggle	0%	**83%**	17%

Table 1. Comparison of user-selected input dimensions for two types of tasks based on our survey. Twist is most appropriate for tasks with continuous input, pull is most appropriate for tasks with discrete input.

CONCLUSION

We have presented the notion of using cords for input. These can sense one or more independent dimensions, including grasp location, twist, bend and pull. To explore this novel approach, we developed a proof-of-concept cord-based sensor, which senses three of these four dimensions: twist, pull, and touch location. This cord can be used as an eyes-free, readily available, and unobtrusive yet expressive input method for mobile devices. We ran a small study to compare each input type (twist, pull and touch location) in terms of selection accuracy and targeting performance. We also provide a set of design recommendations based on results from our study and a survey. We hope our results demonstrate the feasibility and utility of cords as an input device and inspire researchers and practitioners to integrate this sensor into everyday devices and future research projects.

ACKNOWLEDGMENTS

This work was funded in part by grants IIS-0713509, IIS-0803733, and IIS-0840766 from the National Science Foundation and a grant from the Intel Research Council. This project was also supported by the National Science Foundation Graduate Research Fellowship and the ARCS Foundation.

REFERENCES

1. Blasko, G., Narayanaswami, C., and Feiner, S. Prototyping retractable string-based interaction techniques for dual-display mobile devices. In *Proc. CHI '06*, 369-372.
2. Cypress Semiconductor. 2009. CY8C21234 datasheet. Available from: http://download.cypress.com .edgesuite.net/design_resources/datasheets/contents/cy8c21234_8.pdf.
3. Grossman, T., Balakrishnan, R., and Singh, K. 2003. An interface for creating and manipulating curves using a high degree-of-freedom curve input device. In *Proc. CHI '03*, 185-192.
4. Koch, E. and Witt, H. Prototyping a chest-worn string-based wearable input device. In *Proc. WOWMOM '08*, 1-6.
5. Kulik, A., Paneque, D., and Hochstrate, J., Vectorix: A Low-Tech Mechanical Tracking System. In *IEEE VR '04 Workshop: Beyond Wand and Glove Based Interaction*, 25-27.
6. Merlin Robotics. 2009 Merlin Stretch Sensor. Available from: http://www.merlinrobotics.co.uk /merlinrobotics/merlin-stretch-sensors-c-33.html.
7. Rantanen, J., Impiö, J., Karinsalo, T., Malmivaara, M., Reho, A., Tasanen, M., and Vanhala, J. Smart Clothing Prototype for the Arctic Environment. *Personal Ubiquitous Computing,* 2002, 3-16.
8. Spectra Symbol, Inc. 2009. Membrane Potentiometer Datasheet. Available from: http://spectrasymbol.com/typo3/site/en/softpotsplash/technicalspecs/datasheet.html