Rock-Paper-Fibers: Bringing Physical Affordance to Mobile Touch Devices

Frederik Rudeck and Patrick Baudisch
Hasso Plattner Institute
Potsdam, Germany
Frederik.Rudeck@student.hpi.uni-potsdam.de, Patrick.Baudisch@hpi.uni-potsdam.de

ABSTRACT
We explore how to bring physical affordance to mobile touch devices. We present *Rock-Paper-Fibers,* a device that is functionally equivalent to a touchpad, yet that users can reshape so as to best match the interaction at hand. For efficiency, users interact bimanually: one hand reshapes the device and the other hand operates the resulting widget.

We present a prototype that achieves deformability using a bundle of optical fibers, demonstrate an audio player and a simple video game each featuring multiple widgets. We demonstrate how to support applications that require responsiveness by adding mechanical *wedges* and *clamps*.

ACM Classification: H5.2 [Information interfaces and presentation]: User Interfaces: Input Devices and Strategies, Interaction Styles.

Keywords: Reconfigurable; input device; optical fiber; malleable; mobile; tangible; wearable; ubicomp; gesture.

INTRODUCTION
Unlike physical controls, touch pads and touch screens have traditionally been flat and featureless. To reduce error and improve affordance, researchers have proposed adding physical constraints (e.g., *Bricks* [2]).

Rekimoto et al. provide touch screen widgets with better affordance by overlaying interactive screens with application-specific, physical constraints called *DataTiles* [9]. An embossed circular groove, for example, constrains the user's finger to the touch area forming a dial, thereby affording dialing motion.

The vast majority of today's touch screens, however, are used in *mobile* devices. Unfortunately, DataTiles do not transfer to mobile devices. The reason is that tiles require space. In addition, the underlying touch sensitive platform even has to be large enough to provide space for multiple tiles. This limits the approach to tabletop-style devices.

In this paper, we tackle these limitations and demonstrate how to bring custom-shaped physical controls to mobile touch devices.

ROCK-PAPER-FIBERS
The main idea behind *rock-paper-fibers* is to allow users to obtain physical affordance by deforming the device.

Figure 1 shows our prototype, which consists of a bundle of optical fibers held together by a hand piece. The top ends of the optical fibers are touch-sensitive. Functionally, the device is therefore equivalent to a touchpad: it offers a two-dimensional array of touch-sensitive elements. Unlike a regular touchpad, however, each sensor element has been extended using an optical fiber. Since fibers can be bent, this allows users to deform the "touchpad" or break it apart into multiple touch-sensitive elements (e.g., Figure 2).

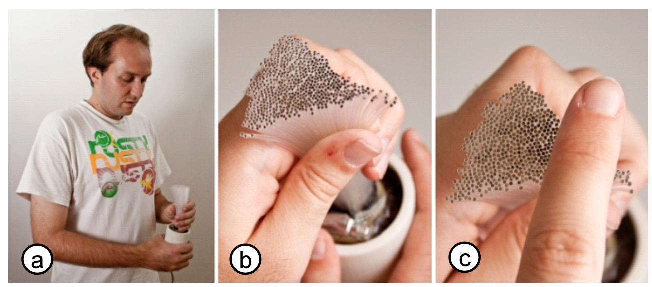

Figure 1: (a) Functionally, *rock-paper-fibers* is equivalent to a touchpad. However, each sensor element is extended using an optical fiber, making the "touchpad" deformable. (b) In order to make the audio player "play", the user reshapes the device into a "▶" symbol and (c) strokes his finger across the "▶". This causes the device to recognize its new shape and execute the play command.

Walkthrough
Figure 1b and c show an example interaction: to make an audio player "play", the user reshapes the device into a "▶" symbol. Stroking across the "▶" causes the device to recognize its new shape and execute the play command.

Figure 2 continues the audio player example. (a) To adjust volume, the user forms a slider by spreading out sensor elements into a strip. (b) The user slides a finger across the strip, which causes the prototype to recognize the slider and define a start and end point. (c) The user continues to drag the finger across the slider, which adjusts the volume. Tapping a position on the slider sets volume directly. (d) To obtain additional precision, the user squeezes the slider harder, making it longer and thus allowing for additional precision. (e) To jump to the next song, the user forms a navigation menu and (f) registers it by stroking across. (g) The user now jumps to the next tracks ahead by repeatedly tapping the next track button. (h) Finally, the user pauses the audio player by forming a "II" icon and (i) activates it by stroking across.

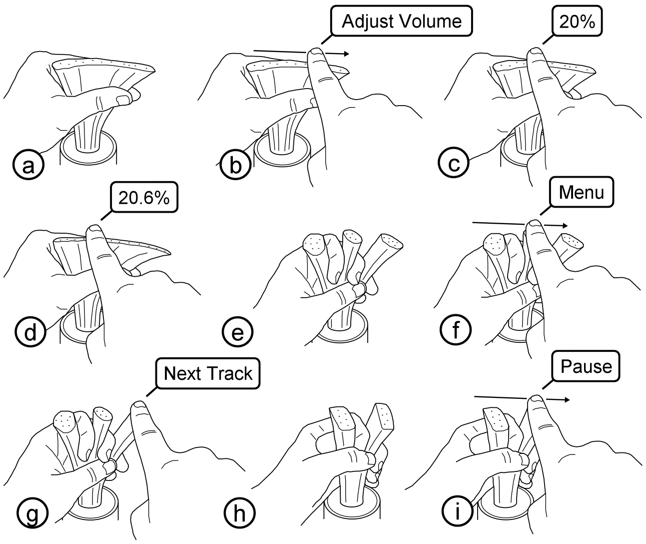

Figure 2: … continuing the audio player scenario from Figure 1, (a-d) the user adjusts audio volume, (e-f) invokes a menu, (g) jumps several tracks ahead, and (h-i) pauses the player.

Benefits and limitations

Rock-paper-fibers brings custom-shaped physical controls to mobile touch devices. To achieve the required smallness (1) users reshape *the device itself* and (2) we serialize the interaction, i.e., users manipulate one widget at a time. The result is a deformable/reconfigurable "touchpad" that manipulate efficiently using bimanual interaction.

Unlike other devices that physically *constrain* the user's fingers, rock-paper-fibers widgets merely *guide* the user. By maintaining physical contact with the protruding fiber bundle users receive continuous tactile feedback, similar to how users find their way around a mouse wheel or a set of physical sliders by maintaining physical contact with it.

Unlike on spacious tabletops, custom-shaped interaction on mobile devices comes at a price. The repeated reconfiguration requires additional manual skills, which limits the range of widgets an individual user is able to produce. It also costs time, as users customize interface elements repeatedly. For applications where responsiveness is *critical,* we therefore allow users to prepare multiple widgets ahead of time (see section "Persisting Widgets").

Contribution

The primary contribution of this paper is the general concept of *bringing physical affordance to mobile touch devices* by making the touch *device deformable.*

We demonstrate this concept at the example of *one possible* form-factor, which we call rock-paper-fibers.

RELATED WORK

Rock-paper-fibers is related to arbitrarily-shaped and deformable touch devices, tangibles, and optical fiber.

Recent advances in touch technology [8] have allowed researchers to touch-enable non-planar shapes [10], such as the grip of a stylus [14]. Harrison et al demonstrated how to touch-enable human skin using acoustic tracking [4].

Several researchers have created Organic User Interfaces, such as *Gummi* [12] and *PaperPhone* [5] that users operate by bending the device itself [13]. Harrison and Hudson created deformable interfaces by combining pneumatics with optical tracking [3]. Schwarz et al. enabled users to interact with the chord of their headsets [11]. Wimmer and Baudisch demonstrated how to touch-enable stretchable fabrics using time-domain reflectometry [18]. Taylor and Bove showed how to change the function of a device without deforming—users instead change the way they hold the device (*Bar of soap* [15]).

To provide physical affordance through specialized physical shape is also the objective of *tangible computing*. The *Actuated Workbench* combines tangible pucks with separate mechanical constraints [7]. *DataTiles* enhance tabletop widgets by overlaying a physical counterpart [9]. *SLAP widgets* extend this to tangibles with moving parts [16].

Early on, optical fiber was used to sense bending in *Datagloves*. Recent projects use optical fiber to redirect optical sensing (e.g., *Flyeye* [17], *Lumino* [1]).

PROTOTYPE

Hardware

Figure 3a shows our main prototype taken apart. The photo reveals a bundle of about a thousand 1mm optical plastic fibers, observed by a 720p webcam (Microsoft *LifeCam Cinema*). A 3D-printed casing holds both parts together. Figure 3b shows a smaller wireless version we created; it uses a 2.4GHz wireless spy cam, powered by a stack of watch batteries. The fiber bundle was repurposed from a fiber optic lamp.

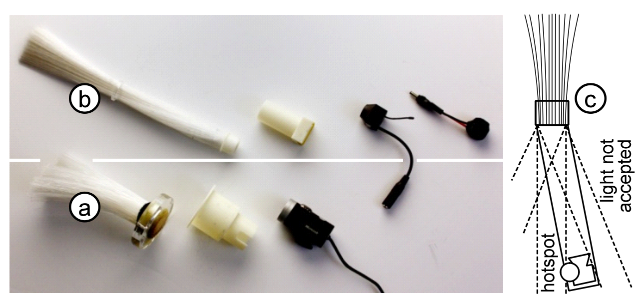

Figure 3: (a) Our prototype consists of a bundle of optical fibers touch enabled by pointing a web cam at the opposite end. (b) Mobile version with wireless camera. (c) Adding illumination.

The device detects touch as follows. Environmental light falls into the optical fiber and is transmitted down the fiber, where it is diffused and observed by the camera. During touch interaction, the user's hand shadows some of the fibers, causing them to appear dark to the built-in camera. The device thus implements a simplified form of front diffuse illumination.

This design is optimized for smallness and mobility. As an alternative, we have complemented prototypes with an IR illuminant, so as to make them independent of environmental light. As illustrated by Figure 3c the illuminant sits next to the camera, sending light through the fiber bundle, where it is reflected by the user's hand. The figure shows how we tilt the handle far enough to get camera and illuminant out of the hotspot, yet not so far as to leave the fiber bundle's acceptance cone.

Registration

Figure 4 illustrates how the device recognizes widgets. Whenever the device is reconfigured, it has no way of knowing how fibers are spatially organized. To reestablish this, (a) users swipe their finger across the device, causing the device to see fibers turn on and off (b). The only meaningful information the device can extract from this is *how many* fibers are covered at a given time (c). Plotting this number over time, however, forms a characteristic pattern. (d) By matching this pattern against a database of labeled widget templates, our prototype identifies the widget. Matches are computed using *Dynamic Time Warping* (the device also supports the *$1 Recognizer* [19]).

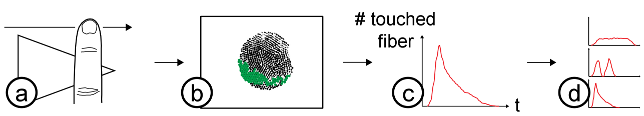

Figure 4: Widget registration: (a) The user slides a finger across the widget. (b) The device counts touched fibers (c) over time and (d) matches the resulting graph with its widget database.

Figure 5 illustrates how we populate the widget database. Swiping the finger across a widget convolves the shape of the finger with the shape of the widget. Consequently, we create the widget graphs in the database by computing this convolution. Alternatively, our prototype also allows adding widget definitions by demonstration.

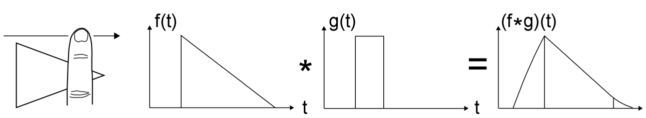

Figure 5: Each graph in the widget database represents the convolution between finger and widget.

The device determines the amount of touched fibers by thresholding brightness. While this can be done on a per-pixel basis, we obtain more reliable recognition by thresholding on per-fiber basis. To enable this, we locate fibers using *Hough circles* (OpenCV [6]). This is done once per lifetime of the device and stored in a calibration file.

Operating spatial widgets, such as sliders and pads

To allow operating a slider using direct touch (as in Figure 2c), rock-paper-fibers needs to determine which fibers correspond to which location. To determine this, rock-paper-fibers records *when* each fiber was occluded during registration; this time corresponds directly to the x coordinate of the respective fiber. When operating a slider widget, the device can now determine the slider's value by averaging the x coordinates of the occluded fibers.

Similarly, rock-paper-fibers allows operating two-dimensional widgets, which we call *pads.* To obtain x and y location, users register pads using a horizontal swipe followed by a vertical swipe. Figure 6 shows the sieve + ring mechanism we use to optimize pads.

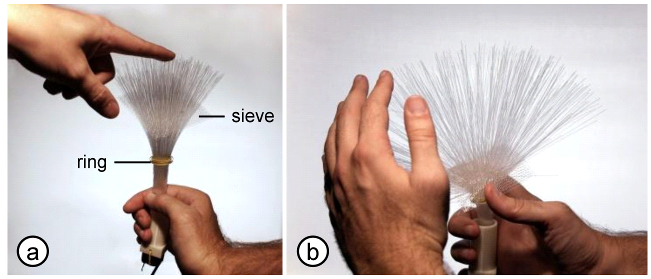

Figure 6: (a) Rock-paper-fibers allows operating two-dimensional widgets. This *pad* was created with the help of a *sieve* that distributes fibers homogenously. (b) Moving the ring towards the sieve spreads fibers out, enlarging the interaction surface.

Application Interface

We envision rock-paper-fibers to be integrated as a stand-alone mobile device, as illustrated by the walk-through. For prototyping, however, we connect the device to a PC. The rock-paper-fibers framework allows us to map widget controls to arbitrary GUI elements using the Mac OS X accessibility API or create predefined keyboard and mouse events by using the Quartz Event Service. To associate a rock-paper-fiber widget with a function in an application program, users pick a rock-paper-fiber widget and press the "s" button on the keyboard while hovering with the mouse over the desired application widget.

PERSISTING WIDGETS USING WEDGES AND CLAMPS

To support applications where responsiveness is of the essence, rock-paper-fibers allows users to configure multiple widgets at once. Since users will not necessarily be able to shape and hold multiple widgets at once, we offer physical constraints.

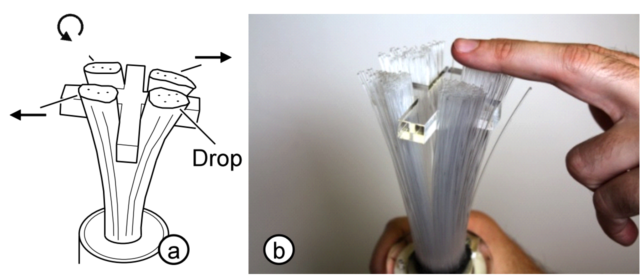

Figure 7: To prepare a game of Tetris, the user splits the device into 4 buttons using a 2×2 wedge.

Figure 7 illustrates the use of a *wedge:* To prepare a game of Tetris, the user has to split the fiber bundle into four buttons by inserting a 2×2 wedge into the device. If an application requires multiple *custom* widgets, such as the interface for a racing game, clamps provide the required flexibility (Figure 8). Clamps are laser-cut from 10mm acrylic and held together using the spring element from a clothes-pin.

In addition to allowing users to create multiple persistent widgets, wedges and clamps also free up the non-dominant hand, thereby allowing users to use both hands to interact with the application, e.g., while resting the device on their lap.

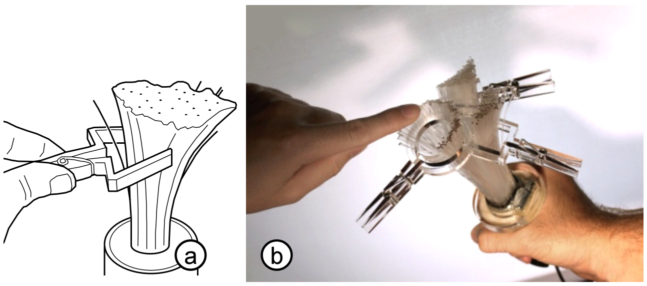

Figure 8: (a) A triangular clamp allows persisting a play button. (b) These three clamps implement the steering dial, gas button, and brake button for a racing game.

RECOGNITION RATE OF FIRST TIME USE

In order to determine the reliability of our widget interaction, we conducted a brief validation. We recruited 9 participants from our institution. After 5 minutes of training, each participant performed each of 7 gestures 3 times by hand in a well-lit room.

Figure 9 shows the resulting recognition rates we received with a 9-fold cross validation. The chart shows that participants performed well with the multi-part widgets, but also that shaping widgets, such as ▶ vs. ◀ , is more prone to misrecognition and requires additional training.

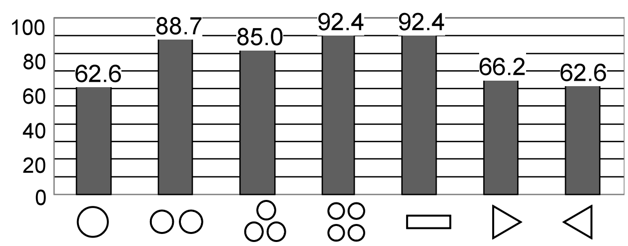

Figure 9: Recognition rates for a set of 7 gestures by first time users.

CONCLUSIONS AND FUTURE WORK

In this paper, we explored how to bring physical affordance to mobile touch devices. The main idea behind our approach is to let users reshape *the touch device itself.* We presented an interactive prototype, with matching sensing mechanism, and algorithm.

As future work, we plan to explore more compact form factors, such as malleable devices touch-enabled using time-domain reflectometry [18] (Figure 10).

ACKNOWLEDGMENTS

We thank Christian Steins, who contributed input on optical fiber and vision algorithms.

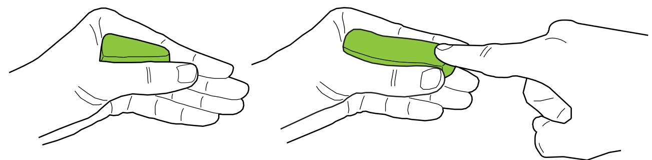

Figure 10: Future form factor touch-enabled using TDR [18].

REFERENCES

1. Baudisch, P. Becker, T, and Rudeck, F. Lumino: tangible blocks for tabletop computers based on Glass fiber bundles. In *Proc. CHI '10,* pp. 1165–1174.
2. Fitzmaurice, W., Ishii, H., and Buxton, W. Bricks: laying the foundations for graspable user interfaces. In *Proc. CHI '95,* pp. 442–449.
3. Harrison, C. and Hudson, S. E. Providing dynamically changeable physical buttons on a visual display. In *Proc. CHI '09, pp.* 299–308.
4. Harrison, C., Tan, D., and Morris, D. Skinput: appropriating the body as an input surface. In *Proc. CHI '10*, pp. 453–462.
5. Lahey, B., Girouard, A., Burleson, W. and Vertegaal, V. PaperPhone: Understanding the Use of Bend Gestures in Mobile Devices with Flexible Electronic Paper Displays. In *Proc. CHI '11,* pp. 1303–1312.
6. OpenCV http://opencv.willowgarage.com
7. Pangaro, G., Maynes-Aminzade, D., Ishii, H. The actuated workbench: computer-controlled actuation in tabletop tangible interfaces. In *Proc. UIST '02,* pp. 181–190.
8. Rekimoto, J. SmartSkin: an infrastructure for freehand manipulation on interactive surfaces. In *Proc. CHI '02*, pp. 113–120.
9. Rekimoto, J., Ullmer, B., and Oba, H. DataTiles: A modular platform for mixed physical and graphical interactions, In *Proc. CHI '01,* pp. 269–276.
10. Roudaut, A., Pohl, H., and Baudisch, P. Touch on curved surfaces. In *Proc. CHI '11*, pp. 1011–1020.
11. Schwarz, J., Harrison, C., Hudson, S., Mankoff, J. Cord input: an intuitive, high-accuracy, multi-degree-of-freedom input method for mobile devices. In *Proc. CHI '10,* pp. 1657–1660.
12. Schwesig, C., Poupyrev, I., Mori, E.: Gummi: a bendable computer. In *Proc. CHI '04,* pp. 263–270.
13. Scott, J., Brown, L., and Molloy, M. Mobile Device Interaction with Force Sensing. In *Proc. Pervasive'09,* pp. 133–150.
14. Song, H., Benko, H., Guimbretière, F., Izadi, S., Cao, X., Hinckley, K. Grips and gestures on a multi-touch pen. In *Proc. CHI '11*, pp. 1323–1332.
15. Taylor, B. T. and Bove, V. M. Graspables: grasp-recognition as a user interface. In *Proc CHI '09,* pp. 917–926.
16. Weiss, M., Wagner, J., Jansen, Y., Jennings, R., Khoshabeh, R., Hollan, J., and Borchers, J. SLAP widgets: bridging the gap between virtual and physical controls on tabletops. In *Proc. CHI '09,* pp. 481–490,
17. Wimmer, R. FlyEye: grasp-sensitive surfaces using optical fiber. *In Proc. TEI '10*, pp. 245–248.
18. Wimmer, R., and Baudisch, P. Modular and deformable touch-sensitive surfaces based on time domain reflectometry. In *Proc. UIST '11*, pp. 517–526.
19. Wobbrock, J., Wilson, A. and Li, Y. Gestures without libraries, toolkits or training: A $1 recognizer for user interface prototypes. *Proc. UIST '07,* pp. 159–168.