Tickle: A Surface-independent Interaction Technique for Grasp Interfaces

Katrin Wolf[1], Robert Schleicher[1]
katrin.wolf@acm.org
robert.schleicher@tu-berlin.de
[1]TU Berlin, Quality & Usability Labs

Sven Kratz[2], Michael Rohs[3]
kratz@fxpal.com
michael.rohs@hci.uni-hannover.de
[2]FX Palo Alto Laboratory, [3]University of Hannover

ABSTRACT
We present a wearable interface that consists of motion sensors. As the interface can be worn on the user's fingers (as a ring) or fixed to it (with nail polish), the device controlled by finger gestures can be any generic object, provided they have an interface for receiving the sensor's signal. We implemented four gestures: tap, release, swipe, and pitch, all of which can be executed with a finger of the hand holding the device. In a user study we tested gesture appropriateness for the index finger at the back of a hand-held tablet that offered three different form factors on its rear: flat, convex, and concave (undercut). For all three shapes, the gesture performance was equally good, however pitch performed better on all surfaces than swipe. The proposed interface is an example towards the idea of ubiquitous computing and the vision of seamless interactions with grasped objects. As an initial application scenario we implemented a camera control that allows the brightness to be configured using our tested gestures on a common SLR device.

Author Keywords
Grasp, back-of-device, rear, swipe, pitch, gesture.

ACM Classification Keywords
H5.2 [Information interfaces and presentation]: User Interfaces. - Input Devices and Strategies.

General Terms
Design, Human Factors.

INTRODUCTION
Most devices offer built-in sensors or buttons for receiving input commands. For gestural interfaces, the sensors are usually motion sensors or cameras. Gesture interfaces do not necessarily have a GUI, which could support gestural guidance. This is a strong argument for establishing similar gesture commands for different devices. Input technology does not generalize across devices. We propose to develop a generic user input device once and then re-use it instead of re-implementing it for each case. We are studying whether an input device for gestural interactions could be re-used, such that sensors (such as motion sensors) need not be built into devices but could rather be worn on the user's hands instead (see Fig. 1). Wireless communication [7] and RFID technology could for instance connect a wearable technology with a desired device. In the field of mobile ubiquitous computing there are still many questions to be answered. In this paper, we address in particular to what extent the form factor of the individual device we want to control needs to be taken into account in order to realize motion-based gesture recognition both on and above the device surfaces. As already known from handles and SLR cameras, ergonomic form factors can stabilize the grasp and thus simplify usage and increase comfort. We developed an interface which aims to interpret gestures that are performed on generic hand-held objects. The question we address with this paper is whether a gestural movement above surfaces of different shapes will affect the gesture

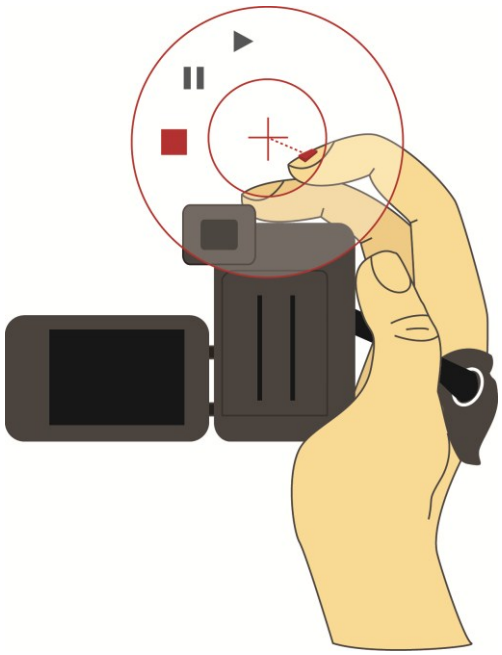

Figure 1. Through finger rings or lacquer, sensors could serve for physically separating the gestural input from the device. We propose a generic input interface that could control any smart object that has some computation facilities. In the shown example, the interface would control a camera.

TEI 2013, Feb 10-13, 2013, Barcelona, Spain.

applicability. Therefore, we investigate our proposed interaction technique on flat surfaces as well as other common shapes, namely convex and concave handles.

This paper is a proof-of-concept rather than a technology paper for a gestural back-of-device interaction for various devices. The general idea is that these finger gestures can be performed while holding the device. Common form factors for such device handles are flat (e.g. tablets) as well as convex (e.g. cameras) and concave curved (e.g. car door handles). Therefore, we discuss existing interfaces and interaction techniques regarding their suitability with respect to different surface form factors on which the gestures are executed.

The paper is structured as follows: after discussing existing work on gestural interfaces regarding their suitability for being scaled to control different devices in changing environments, we propose *Tickle*, a finger-worn interface for gestures that can be executed on the back of devices with different form factors. We explain our interaction concept as well as the interface prototype, and present its evaluation in a user study. We conclude by discussing the scalability of the emerging design space of device-independent user interfaces.

RELATED WORK

We distinguish between three locations where user interface sensors can be placed for measuring gestural input signals: the environment, the device, and the user (e.g. wearable interfaces). We introduce relevant work in all three interface categories and show how our project differs from existing work and what lead us to the *Tickle* design.

Environment-placed sensors

For sensing surface gestures, cameras can be set up in the environment. The resolution of novel depth camera systems [16] allows finger gesture classification, at least for distinguishing how many fingers per hand are stretched or bended. Although the technology is stable in well-illuminated rooms, it does not work well under variable lighting conditions or when the grasped device or other objects occlude the fingers. As we are interested in interactions with mobile devices, sensors that are statically installed in the environment would limit our use cases substantially.

Hand-held devices with included sensors

Roudaut *et al.* [17] investigated the touch inaccuracy of touch input on differently curved shapes using a camera that was placed in a box underneath the surface and is related to our work as they investigate performance on different shapes. *LucidTouch* [26] is a proof-of-concept work for back-of-device interaction using a camera attached to the device. The noticeable camera protrusion on the hand-held device could be a usability issue for this prototype. *NanoTouch* [4] solved that problem through using a capacitive surface on the device's rear. Steward *et al.* [23] and Shen *et al*. [19] glued two iPhones, and Wolf *et al.* [27] two iPads together at their rear and thereby got a prototype with a capacitive surface on the front and on the back for exploring the extended gestural design space of two-sided touch-based interactions with mobile devices. Others investigated the usage of different sensors for extending the interaction possibilities with mobile devices. *HoverFlow* [12] used IR sensors, which allow hover gestures. *Touché* [21] allows the number of touching fingers to be identified or whether the whole hand is grasping, by measuring capacitive response of object and human at multiple frequencies. The current implementation of *Touché* does not distinguish between gestures that are performed with the touching fingers, such as swipe or pitch.

While the variety of available sensors can extend the common device interaction space remarkably, the interface design in most (except *Touché* [21]) cases lack scalability, which means that they are limited to a particular setup and cannot be transferred to other devices or scenarios that easily. *Touché* [21], can augment any object with an electrostatic surface, but aims at grasp detection rather than identifying movement-based finger gestures. Body extensions and wearable interfaces, which work with any graspable smart object, of course require augmenting these objects with a wireless communication unit; however the objects themselves would not require a certain form factor or surface design that is driven by the gesture sensing technology because the sensors are worn on the user's fingers.

Body-worn sensors

WristCam [25], *PinchWatch* [13] and *ShoeSense* [3] all use cameras for finger gesture sensing. This technology has to deal with the usual problems of camera signals (light conditions, gesture occlusion) and is therefore hardly suitable for sensing finger movements when the hand is grasping a device. *Lightglove* [8] has built-in optical detectors (that are, like in *WristCam* [25], integrated in a wrist band) for measuring LED light that is reflected by the fingers. Again, this setup requires users to have hands "free".

Harrison *et al.* [6], Saponas *et al.* [19], and Rekimoto [15] measure natural finger gestures using various signals, such as acoustic signals, electromyograms (EMGs) via electrodes that record forearm movements. *MagiTact* [11] uses the built-in magnet sensor of a phone for detecting hand gestures through moving a magnet around the device. That magnet is grasped in the prototype but can be included in a ring worn on the finger which then would allow free hand usage and holding devices. *Nenya* [2] uses magnet sensors for sensing (with a wrist-worn magnet sensor) finger ring rotations around the finger that requires the "free" hand to rotate the ring. Magnet sensors allow for movements of magnet or metal in a magnetic field to be detected, but for sensing accurate finger movements, motion sensors seem to be more promising, because the signals of magnet sensors might be too ambiguous due to

other surrounding magnet field of the earth and other nearby magnets. In addition, magnetometers can only track a single point in motion. Amma *et al.* [1] detect whole-hand gestures for handwriting by attaching accelerometers and gyroscopes at the palm. Finger gesture recognition again cannot be provided because the sensors are placed at the palm. The *Body Coupled FingeRing* [5] allows for discrete commands (finger-tip typing actions) to be detected using an accelerometer, and *Ubi-Finger* [24] uses bend sensors and accelerometers for measuring finger movements to control continuous parameters, such as scroll bar or volume. In contrast to the related work reported so far, the proposed *Tickle* interface is designed for different objects that are represented through different form factors.

TICKLE DESIGN

The design of our interface is driven by the desire to have a situation-independent as well as device- (that is grasped for being controlled) independent interface. The vision leading our design is to develop an interface for interacting with generic smart objects that are held in the hand. Therefore, our design section consists of two sub-sections: first, we present a selection of gestures that are promising for work on any grasped device's surface, but that also allow a wide range of commands, and second, we present a prototype implementation that allows for the concept we propose to be evaluated in a user study.

Gesture set

In a previous study we examined which gestures are promising for generic usage, i.e., gestures that can be executed with a device-grasping hand. We found finger taps and slide movements to be feasible with the index and middle finger for the main grasp types [28]. These two finger movements, as well as further combinations and modifications of them, serve as design units for the gesture commands that we implemented in the current prototype. We implemented four gesture commands: tap, release, swipe, and pitch [18] (see Fig. 2).

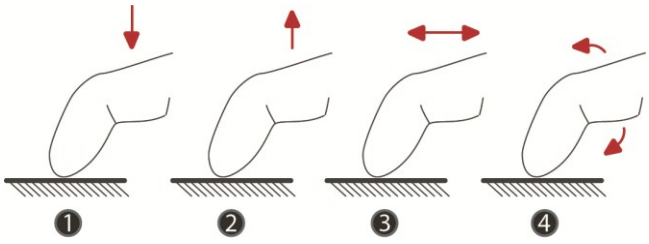

Figure 2. 1) tap 2) release 3) swipe 4) pitch gesture that are measured through finger worn sensors, as drawn in here in red.

Swipe is an accelerative movement done on the surface. *Pitching* a finger is executed through rolling a tapped finger above its tip (see Fig. 2). Tap and release are well-suited to indicating the beginning and end of a command. We implemented two discrete commands for initiating gesture

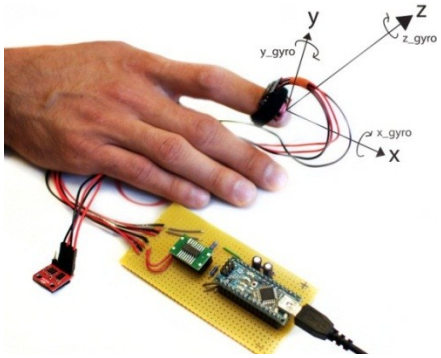

Figure 3. Prototype with IMU digital combo boards that are fixed on fingers with Velcro tape.

mode, i.e. triggering the system to read the finger movement in between the clutching trigger events as actual gesture. These trigger commands are tap and release. After tapping, the gestures for actual slider control are swipe and pitch. The pitch gesture is controlling a slider through direct mapping of the pitch angel modification on a slider value (position control) until a release event is executed as end-of-gesture. The swipe gesture causes a slider to increase or decrease, depending on the direction of the gesture, where the slider movement can be stopped through the second trigger event, which is a tap with immediate release. We implemented these trigger or *clutch* events to avoid the Midas touch problem [9,10]. Without them, subconscious and unintended movements, which are not meant to be a gesture but have a similar trajectory, could be falsely classified as gestures and result in errors and frustration. For starting gesture classification algorithms, push-to-gesture buttons are often used. In our case we were looking for something more subtle. Therefore we defined clutch events (tap, release), which can seamlessly be continued by the actual gesture (swipe, pitch). The clutch events trigger the system to classify the gesture in between.

Prototype

Our prototype consists of two IMU Digital Combo Boards[1] with six degrees of freedom through a gyroscope (ITG3200) and an accelerometer (ADXL345), which communicate over an I2C interface with a wrist-worn Arduino Nano V3 (see Fig. 3). We propose using acceleration sensors in combination with gyroscopes for detecting discrete as well as continuous commands as they are smaller and less impairing than bend sensors and could therefore easily be included in jewelry. Moreover, this sensor combination is very promising for an even more nuanced gesture classification than the tap recognition done by Fukumoto *et al.* [5] through just using acceleration sensors. The current prototype is powered through a USB connection with a PC. Using a battery would be a better option for further versions.

[1] SparkFun Electronics Item Nr. SEN-10121.

The gesture classification is written in Processing. The gesture classification is realized in four steps as described in the following and allows real-time gesture classification.

G force subtraction

We generally subtract the gravity (g-force of 9.81 m/s²) from the acceleration signals to analyze the actual finger movements at any hand orientation. Otherwise, if the acceleration of *g* would be included within the gesture signals, it would be harder to classify a gesture at any possible hand orientation. For subtracting the g-force, we use a high pass filter with a cutoff frequency of 0.15 Hz.

Tap & Release Recognition

We classify a tap in two steps. At first, the acceleration has to be above a defined threshold [14]. The pilot study default value was ½ g force or 4.955 m/s². In the experiment application the threshold was obtained via individual calibration by asking users to provide sample tap gestures, because we observed in a pilot study that users' tap accelerations vary a lot. Secondly the acceleration value that is above the threshold has to decrease under the threshold again after 120 ms. Otherwise the event is not a tap.

An abrupt stop of movement stands for a tapping finger or a finger that was rapidly releasing the back-of-the device.

We distinguish between these two events, the tap and the release, through analyzing the y-value of the gyroscope as it tells about a tilt of the finger tip. If the y value increases, a tap (fingertip movement towards the device) event will be classified; if it decreases the movement will be read as release (fingertip movement away from the device).

Swipe & Pitch recognition

We implemented two optional gestural interaction techniques for continuously controlling sliders, a swipe gesture for rate control and a pitch gesture for position control. *Rate control* means to set the direction of change, to reach a certain position, whereas *position control* allows specifying the target position directly. After the clutch event (tap) is recognized, two algorithms serve for classifying swipe and pitch.

The swipe and pitch gestures are both classified based on y-value of the gyroscope. As in most interfaces (e.g. browsing through the contacts on an iPhone), the swipe gesture was used for rate control. We defined a threshold of 450 pixels/s, which corresponds to 31.3°/s (degrees per second) for swipe as absolute value. The slider was moved in the same direction as the swipe that was performed, i.e. a positive value will result in slider movements to the right. A further tap event will be interpreted as end-of-swipe clutch and therefore stop the slider change.

The pitch gesture was mapped onto position control. A change in pitch / finger angle of 6.95°/s (or 0.1° because the function was called once every 15 ms) was mapped to a 1 pixel increment/decrement of the slider that was 840 pixels long in total. Sliding over the whole slider distance therefor required a finger tilt of 84° in total. For having an intuitive mapping, we changed the slider according the finger pitch and let increasing pitch angles move the slider to the left and decreasing angles to the right. The end-of-gesture clutch for pitch was a finger release from the surface.

Figure 4. Participants while performing a gesture at the undercut shape.

Sensor placement

In a pilot study we tested the three finger segments and gained the best gesture classification results when the sensors were worn at the fingertip segment. The middle and palm-nearest segments do not represent the movements clearly enough, at least for our algorithm. The sensors were fixed with Velcro on the first finger segment of our volunteers and the Arduino with the circuit board was worn on the wrist (see Fig. 4).

USER STUDY

Experiment design

Our user study had a 2x2x3 within design for comparing the scalability of the *Tickle* interface for recognizing two continuous control techniques. The factors were: 2 types of control (rate control via swipe, position control via pitch) in 2 directions (i.e. increasing or decreasing a value) for back-of-device interactions on 3 surface shapes (flat, convex, concave, see Fig. 5) on those the gestures are performed. Our baseline was a flat surface as that represents most current touch-based interfaces. We did not include the most widespread touch sensing technology of capacitive device surfaces as we intended to adhere to our idea of worn sensors.

As dependent variables we measured subjective effort (for each surface interaction-technique combination) using the SMEQ scale, because it is known to be very sensitive with small sample sizes [20], the interaction time for each trial, and the offset error over the whole trial for the slider configuring gestures (swipe, pitch). The last three performance variables were recorded in log-files during the experiment. A trial started after the new target position of the slider was presented with a button click of the participants and was finished when the participants successfully had placed the slider bar at the given position.

Surface shapes

The basic shape we used as control condition was flat (see fig. 5.1) like most mobile devices such as phones and tablets, especially if they have no (or few) physical buttons but a touch sensitive screen.

We compared the flat backside with 2 curved shapes regarding the ability to perform touch-based gesture on it: a convex and a concave (see fig. 5.2-3) shape. Many ergonomic handles and form factors are concave in order to fit well in users' hands, e.g., a SLR camera or door handles. Concave form factors are for example used for car door handles and allow for the easy opening a door by pulling it. Both curved surfaces have function-dependent benefits and are ubiquitous in ergonomic product design.

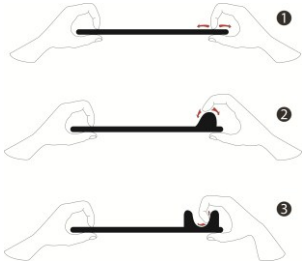

Figure 5. Surfaces swipe and pitch gestures were executed on: 1) flat 2) convex 3) concave (undercut).

We extended a tablet with a lightweight concave and convex extension in order to test the shape effect on gesture performance, without influencing other factors such as weight or the way to grasp the device too much. The size of the convex shape was inspired by the convex shape of SLR cameras that allow for grasping the device ergonomically. The undercut was just the inverse form of the convex extension.

Research questions

We were questioning if users are able to perform both gestures (swipe, pitch) equally well at any of the three test shapes (flat, convex, concave). Furthermore, we were investigating whether both gestures differ in performance.

Procedure

12 participants (11 right-handed and 1 ambidextrous, 11 males, 1 female, aged 29.3 years on average, ranging from 24 to 37 years) were asked to execute the two interaction techniques (swipe, pitch) on all three surfaces (flat, convex, concave) for moving a slider from a changing default position to a predefined destination. The default and destination value of the slider were chosen in order to force the participants to move the slider more or less the same distance in both directions over all trials: away from the hand palm or towards it. The order of the interaction techniques and surface shapes was randomized. For each of the 2x3 conditions, in the following also called *trial blocks*, 10 trials were completed, resulting in 60 trials per person. In the beginning, we calibrated the thresholds for the participants' tap and release events. Before any task trial, we had a training trial lasting about two minutes, which was completed as soon the participants felt comfortable with the interaction technique and task.

RESULTS

User performance was determined as interaction time and slider target approximation (offset over time) for each condition. The offset over time is the size of the area between the actual slider position and the target (see Fig. 6) that had to be reached for ending one trial.

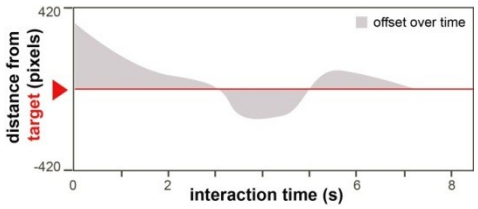

Figure 6. Slider offset over interaction time for each user and trial serves to compare the efficiency.

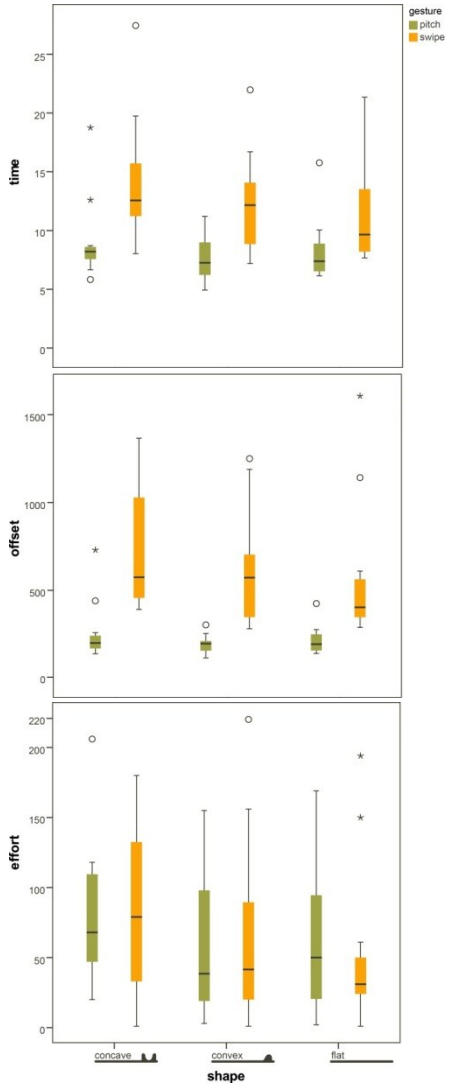

Figure 7. (top) Interaction time per condition (center) Offset over time per condition (bottom) Perceived effort per condition.

Repeated measure ANOVAs with *gesture* and *shape* as within-subject factors using a 5% significance level showed a significant difference for the dependent variable time ($F_{1,35}$=38.278, p<0.001) and for slider offset ($F_{1,28}$=59.850, p<0.001) only for the factor *gesture*. Thus, while the type of gesture (swipe or pitch) had a significant influence on both performance parameters (see Fig 7), the shape the gesture was performed on did not. Regarding the shape, no significant results were observed for time and offset error (time: $F_{2,52}$=1.364, p=0.265, offset: $F_{2,51}$=0.935, p=0.399). There was no significant interaction of gesture*shape ($F_{2,598}$=1,325, p=0.267), which would otherwise indicate that a certain gesture can be performed better on a certain shape.

We found a significant difference of required interaction time for the different trial numbers (time: $F_{8,246}$=3.232, p=0.002) within a block with later trials requiring less time (see Fig. 8). We will interpret this as a learning effect in the discussion. No effect could be observed for the offset ($F_{8,265}$=1.695, p=0.123).

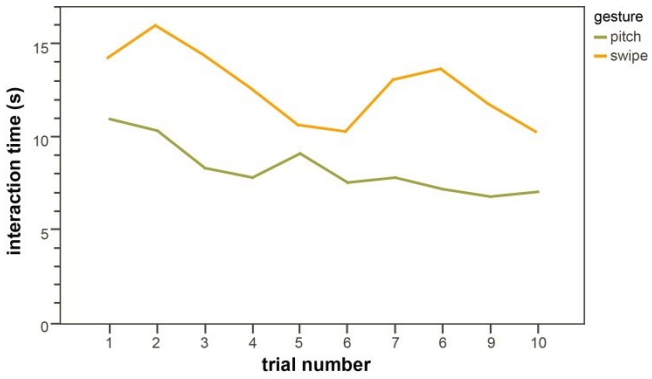

Figure 8. Average of time per trial over all participants.

The experienced effort measured with the SMEQ scale did not show any significant difference at all (*shape*: $F_{2,56}$=0.948, p=0.394, *gesture*: $F_{1,48}$=0.01, p=0.971, *shape***gesture*: F…). Thus all interface variants were perceived the same way (see Fig. 7).

DISCUSSION

Our results show that neither quantitative performance nor experienced effort was significantly different between the surface shapes flat, convex, and concave (undercut). Regarding our second question, we found that, surprisingly, the gesture type pitch resulted in significantly shorter interaction time and less offset than the swipe gesture. These results will be discussed in detail below.

Gesture

In general, pitch performed better than swipe. A reason for this better performance might be that the constant contact with the surface while pitch allows for a continuous adjustment of the current movement. For swipe the finger is moving above the surface, which might provide less support for motor control. Nevertheless, the pitch-gesture turned out to be consistently more efficient with regard to time and movement offset.

For the swipe gesture with a rate control, some participants (3) mentioned during the experiment that they would prefer if the slider changes faster when the gesture (swipe) is executed with high acceleration and slower for a slow swipe. The higher interaction time for swipe could be reduced if the gesture implementation considered this aspect. Moreover, we observed that correcting the slider position with the swipe gesture was really hard for little corrections and lead almost every time to overshoots, which affected the execution time and the offset error.

One third of the participants mentioned that swipe was not perceived as being equally feasible for both directions, especially on the undercut shape: swiping away from the hand palm through stretching the finger or towards it through bending the finger. Three subjects found bending the finger to swipe easier than stretching it. One subject found it the opposite. A reason for the lower feasibility of bending a finger in the undercut shape might be that the sensors sometimes got caught on the surface. We are aware that a finger ring or our Velcro prototype that is worn on the fingertip segment is somewhat lacking in comfort and propose to build an alternative smaller sensor board and glue it on the finger nail with nail-polish for a more advanced prototype.

Another reason for the lower performance of swiping might have been that sometimes the trigger gesture (tap) automatically resulted in a swipe gesture. That effect is known as the Midas touch problem and an attempt to overcome this was made through a recalibration for certain user. In our observation the problem occurred rarely, but additional improvements might reduce errors further. The unintentional events might have affected the gesture performance, which would be a bias over all tested shapes and therefor would not decrease the validity of our results regarding the influence of different shapes. In a similar vein, there was no interaction of gesture*shape on performance parameters, i.e. no accumulation of such misclassification for a certain shape.

Both gestures (pitch, swipe) showed an increasing performance over the trial numbers. This we interpret as a learning effect due to the fact that touch-based gestures for hand-held consumer products are usually performed on their front (except Sony Vita) and users are not familiar with back-of-device gestures. Moreover, one participant had the impression that the gesture classification improved over time. We suspect that participant had a reasonable learning curve himself; but attributed this improvement to a learning system.

Shape

The shapes neither affected the performance nor the perceived effort but did affect the preference as noticed in comments (between the trials) in the experiment. The

participants tended to like the convex shape more than the flat one and definitely found the undercut weird for executing gestures in it. This is in line with common ergonomic form factor design for devices that are held like the tablet in our experiment. Camera devices for example, which were the inspiration for choosing the convex shape, have an ergonomic design that is similar to our prototype.

Even though no performance differences were found between all three shapes, half of the participants mentioned that moving a finger above the undercut for executing the swipe is more difficult than pitching at the undercut or swiping at the other shapes (flat, convex), because the undercut sometimes hinders the movement. Usually, an undercut is used instead to apply pull forces, such as for opening doors or for gripping on to climbing holds. Therefore, we suggest that a convex surface might be the better choice for grasp-based interfaces.

Transferability

With our experiment, we provided first evidence that our proposed tickle design might work for differently shaped surfaces. After showing that our interaction design tends to be shape-independent, we want to explore more parameters that increase the degree of device independence, but also in order to offer the possibility to perform input gestures with more than one finger. For instance, using the index and middle finger for different commands could increase the interaction complexity and allow controlling more parameters at the same time.

Therefore, we implemented a camera application for an Andoid phone and stuck the device on the user-facing side of a SLR camera, just at the place where usually the camera display is (see Fig. 9). The phone was arranged on the camera in a way that the phone camera was not occluded through the SLR camera and that the phone screen replaced the SLR camera screen. The sensor that we used in the experiment was now worn on the index as well as on the middle finger. That allowed controlling two parameters of the Android camera application. We chose to enable controlling the picture brightness via graduation curves separately for lights and shadows, like it is usually just possible in picture editing software, such as Photoshop. The index finger controls the lights and the middle finger the shadows. That allows fast and seamless picture configuration under difficult light conditions.

The Tickle interface receives the motion-based gesture commands of two fingers for separately configuring the brightness and contrast of a picture that is dynamically previewed on the Android phone's screen. The implementation seems to be really promising as it worked in informal tests with our colleagues. Together with the fact that cameras running the Android operating system have

Figure 9. Tickle interface implementation on an Android phone for testing 2-finger camera control on a real SLR form factor.

been released recently[2], new possibilities of cross-device interaction arise.

CONCLUSION

We introduced *Tickle*, a grasp interface for generic objects worn on the index finger and showed in a user study that this interface worked on different surface shapes without affecting the gesture performance. Users were able to perform pitch and swipe gestures independent of the surface shape, even though our results showed that both gestures were performed differently regarding interaction time and target approximation. Pitch control, a gesture rarely used in HCI so far (the Lenovo trackpoint being an exception), turned out to be a useful means to execute gesture during grasps in our experimental setup. To demonstrate the potential of Tickle for real-life application scenarios, we implemented a camera control application that allows two-finger gestural control on a SLR device.

In future work, we will, with the aim of being generic in our approach, focus on testing further dimensions regarding the use-case scalability of our design. We will investigate devices that vary in weight and the way that they can be grasped.

ACKNOWLEDGMENTS

We thank Mathias Wilhelm and Christian Kuster for help on implementing the prototype, and all participants of our studies for their time.

REFERENCES

1. Amma, C., Georgi, M, Tanja Schultz, T. 2011. Airwriting: Hands-free Mobile Text Input by Spotting and Continuous Recognition of 3d-Space Handwriting with Inertial Sensors. In *Proc. ISWC 2012*, 52-59.
2. Ashbrook, D., Baudisch, P., White, S. 2011. Nenya: subtle and eyes-free mobile input with a magnetically-tracked finger ring, In *Proc. CHI 2011*, 2043-2046.

[2] Nikon Coolpix S800c, Samsung GALAXY Camera, Polaroid SC1630.

3. Bailly, G., Müller, J., Rohs, M., Wigdor, D., Kratz, S. 2012. ShoeSense: A New Perspective on Hand Gestures and Wearable Applications**,** In *Proc. CHI 2012*, 1239-1248.
4. Baudisch, P., Chu, G. 2009. Back-of-device interaction allows creating very small touch devices. In *Proc. CHI 2009*, 1923-1932.
5. Fukumoto, M. and Tonomura, Y. 1997. 'Body coupled FingeRing': wireless wearable keyboard, In Proc. *CHI 1997*, 147–154.
6. Harrison, C., Tan, D. Morris, D. 2010. Skinput: Appropriating the Body as an Input Surface, In *Proc. CHI 2010,* 453-462.
7. Hinckley, K., Ramos, G., Guimbretiere, F., Baudisch, P., and Smith, M. Synchronous Gestures for Multiple Persons and Computers. In *Proc. UIST 2003*, 149-158.
8. Howard, B, Howard, S. 2001. Lightglove: Wrist-Worn Virtual Typing and Pointing, In *Proc. ISWC 2001,* 172-173.
9. Istance, H., Bates, R., Hyrskykari., Vickers, S. 2008. Snap clutch, a moded approach to solving the Midas touch problem. In *Proc. Symposium on Eye tracking research 2008*, 221-228.
10. Jacob, R.J. 1990. What You Look at is What You Get: Eye Movement-Based Interaction Techniques, In *Proc. CHI 1990,* 11–18.
11. Ketabdar.H., Yüksel, K., Roshandel, M. 2010. MagiTact: interaction with mobile devices based on compass (magnetic) sensor. In *Proc.* IUI 2010, 413-414.
12. Kratz, S., Rohs, M. 2009. HoverFlow: Expanding the Design Space of Around-Device Interaction. In *Proc. Mobile HCI 2009,* 31-38.
13. Loclair, C., Gustafson, S., Baudisch, P. 2010 PinchWatch: A Wearable Device for One-Handed Microinteractions, In *Proc. MobileHCI 2010 Workshop on Ensembles of On-Body Devices*, 4 Pages.
14. Pan, J., Tompkins, W. 1985. A Real-Time QRS Detection Algorithm, In *IEEE Transactions on Biomedical Engineering, Vol. BME-32, No. 3. 1985,* 230-236.
15. Rekimoto, J. 2001. GestureWrist and GesturePad: Unobtrusive Wearable Interaction Devices, In *Proc. ISWC 2001*, 21-27.
16. Ren, Z., Yuan, J., Zhang, Z. 2011. Robust hand gesture recognition based on finger-earth mover's distance with a commodity depth camera. In *Proc. MM 2011*, 1093-1096.
17. Roudaut, A., Pohl, H., Baudisch, P. 2011. Touch input on curved surfaces. In *Proc. CHI 2011,* 1011-1020.
18. Rogers, S., Williamson, J., Stewart, C., Murray-Smith, R. 2011. AnglePose: robust, precise capacitive touch tracking via 3d orientation estimation. In *Proc. CHI 2011*, 2575-2584.
19. Saponas, S., Tan, D., Morris, D., Balakrishnan, R., Tuner, J., Landay, J. 2009. Enabling Always-Available Input with Muscle-Computer Interfaces, In *Proc. UIST 2009,* 167-176.
20. Sauro, J., Dumas, J. 2009. Comparison of Three One-Question, Post-Task Usability Questionnaires, *In Proc. CHI 2009*, 1599-1608.
21. Sato, M., Poupyrev, I, and Harrison, C. Touché: Enhancing Touch Interaction on Humans, Screens, Liquids, and Everyday Objects. In *Proc. CHI 2012*, 483-492.
22. Shen, E. E., Tsai, S. D., Chu, H., Hsu, Y. J., Chen, C. E. 2009. Double-side multi-touch input for mobile devices. In *Proc. CHI 2009*, 4339-4344.
23. Stewart, C., Rohs, M., Kratz, S., Essl, G. 2010. Characteristics of pressure-based input for mobile devices. In *Proc. CHI 2010*, 801-810.
24. Tsukada, K., Yasumura, M. 2002. Ubi-Finger: Gesture Input Device for Mobile Use, In *Proc. APCHI 2002*, 388-400.
25. Vardy, A., Robinson, J., Cheng, L. 1999. The WristCam as Input Device, In *Proc. ISWC 1999*, 199–202.
26. Wigdor, D., Forlines, C., Baudisch, P., Barnwell, J., Shen, C. 2007. LucidTouch: A See-Through Mobile Device, In *Proc. UIST 2007*, 269-278.
27. Wolf, K., Müller-Tomfelde, C., Cheng, K., Wechsung, I. 2012. PinchPad: performance of touch-based gestures while grasping devices. In *Proc. TEI 2012,* 103-110.
28. Wolf, K., Naumann, A., Rohs, M., Müller, J. 2011. Taxonomy of microinteractions: defining microgestures based on ergonomic and scenario-dependent requirements. In *Proc. INTERACT 2011*, 559-575.