PenShaft: Enabling Pen Shaft Detection and Interaction for Touchscreens

Junwei Sun
junwei.sun@huawei.com
Huawei HMI Lab
Canada

Margaret Foley
mjfoley@uwaterloo.ca
University of Waterloo
Canada

Qiang Xu
qiang.xu1@huawei.com
Huawei HMI Lab
Canada

Chenhe Li
chenhe.li@huawei.com
Huawei HMI Lab
Canada

Jun Li
jun.li3@huawei.com
Huawei HMI Lab
Canada

Pourang Irani
irani@cs.umanitoba.ca
University of Manitoba
Canada

Wei Li
wei.li.crc@huawei.com
Huawei HMI Lab
Canada

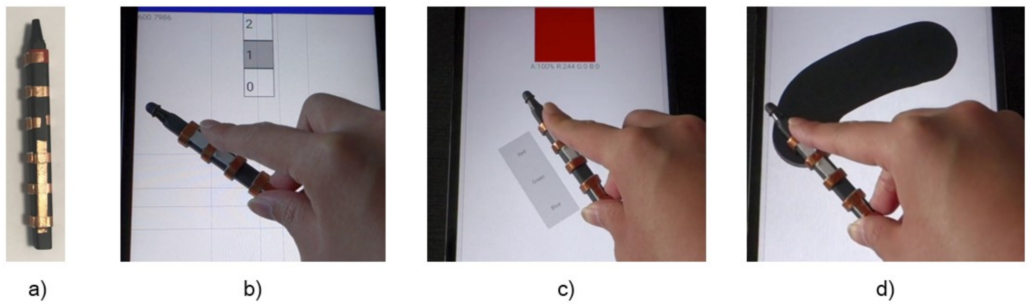


Figure 1: a) Capacitive materials are applied to a pen shaft to create a PenShaft device. b) When the pen shaft is placed on a touchscreen, the tablet can detect the pen. In a drawing application, PenShaft can be used c) to change the pen stroke color by rotating the pen and d) as a brush.

ABSTRACT

PenShaft is a battery-free, easy to implement solution for augmenting the shaft of a capacitive pen with interactive capabilities. By applying conductive materials in a specific pattern on the pen's shaft, we can detect when it is laid on a capacitive touchscreen. This enables *on-pen* interactions, such as button clicks or swiping, and *whole-shaft* interactions, such as rotating and dragging the stylus. PenShaft supports at least six interactions, freeing the pen from having to interact with the layers or menus of a conventional user interface. We validate a device's capability to detect these six interactions with all but two interactions achieving a success rate above 95%. We then present a series of applications to demonstrate the flexibility and utility of these interactions when using the pen.

CCS CONCEPTS

• **Human-centered computing → Touch screens**.

KEYWORDS

pen shaft, sensing, touchscreen

ACM Reference Format:
Junwei Sun, Margaret Foley, Qiang Xu, Chenhe Li, Jun Li, Pourang Irani, and Wei Li. 2021. PenShaft: Enabling Pen Shaft Detection and Interaction for Touchscreens. In *12th Augmented Human International Conference (AH2021), May 27–28, 2021, Geneva, Switzerland*. ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3460881.3460934

1 INTRODUCTION

The pen allows for precise and natural interactions with a touchscreen for tasks such as writing or drawing. However, current applications require additional graphical user interface widgets for changing attributes (e.g. stroke thickness and colour) or switching between modes, which can be slow and error-prone [12, 34]. Popular methods for increasing the pen's interactive capabilities include bimanual pen and touch input [4, 15, 21, 40, 41], augmenting the pen with auxiliary sensing capabilities, such as tilt or pressure [26, 37], and detecting grip changes in the background for adjusting the UI to the user's tasks [14, 31–33]. But such enriched capabilities are only possible with *active* pens (such as the Apple Pencil [1] and Surface Pen [22]), which are powered and operate on a specific platform. These features are absent in *passive* pens, which are unpowered and can work with any capacitive touchscreen.

We introduce PenShaft: a battery-free solution for detecting and interacting with a pen shaft on a touchscreen. Few works have previously studied using the pen shaft for input on a touchscreen. With

PenShaft, the functionality of a passive capacitive pen is greatly augmented to enable whole-shaft interactions, such as using the pen to crop and reorient an image, and on-pen interactions, such as swiping along the pen shaft to change input modes or invoke hotkeys. Since the pen shaft is a physical object, eyes-free interaction is also possible. The pen can be further used as a brush or eraser in a drawing application (Figure 1d).

These interactions, typically unavailable on a passive capacitive pen, are enabled by using conductive materials to facilitate *feedthrough capacitance*. To transfer input from the pen's shaft to the touchscreen, we wrap conductive materials around a regular pen with at least one flat surface on the shaft to create "contact points". These points are connected with another strip of conductive material. When the pen is placed on a capacitive touchscreen while being touched, these contact points appear as multiple touch events in a straight line, acting as a *tag* and allowing the system to detect the pen shaft (Figure 1a). Additionally, "buttons" can be constructed by adding contact points independent from the tag points, which enable clicking and swiping interactions (Figure 1b).

A 6-participant experiment was conducted to validate our PenShaft design and its interactions, finding that all but two interactions achieved a success rate above 95%.

Our contributions include a) a battery-free method to enhance a passive capacitive pen by instrumenting the pen's shaft for whole-shaft and on-pen interactions; b) a validation of these interactions in a user study; and c) a demonstration of applications that incorporate and benefit from PenShaft.

2 RELATED WORK

PenShaft draws inspiration from techniques that augment the pen with novel sensing capabilities, and from touch-enabled input devices. We also discuss methods that have employed *feedthrough capacitance* sensing to enhance tangibles for touchscreens, as PenShaft also uses this capability to enable detection and interaction.

2.1 Pen-Based Interactions

There is a significant amount of research into enhancing pen-based interactions.

2.1.1 Pen Shaft Interaction. Researchers have developed means for augmenting the pen shaft with sensing capabilities to enable novel interaction techniques. In one collection, the pen shaft was designed to detect grip changes [14, 31–33] which could then be used to infer a user's action and enable mode shifts or invoke menu commands. These works focus on implicit detection or background sensing [41], but do not take explicit advantage of the sensing on the pen shaft to control an application's interactive features.

We are aware of three works that examine explicit pen shaft control. Zhang et al. also implemented an input when the device is laid on a tablet to display pen settings [41]. Though it is not a pen-like device, Conté [8, 35], a device inspired by the Toolstone [28], enables interactions where the device is laid flat on a touchscreen to lay guidelines or control menus. These augmentations require a power source, and are not easily adaptable to passive capacitance pens. They also do not explore the performance of explicit pen shaft control as we do, nor do they instrument a pen shaft in the same manner as PenShaft.

2.1.2 Enhancing Pen Sensing Capabilities. Beyond enhancing the pen shaft, researchers have identified many means of augmenting the pen's degrees-of-freedom with auxiliary sensing methods, such as pen tilt [37], pressure [26], roll [2], or by using two or more sensing modalities [13], such as pressure and tilt in conjunction [1]. Xu et al. [38] demonstrated the use of voice input on a pen to control its features while Fellion et al. proposed FlexStylus, which supports rotational deformation of the pen, making it easier to create digital art [9]. Pen interaction has also been enhanced with haptic feedback [7, 19].

2.1.3 Pen + Touch Interaction. Combining pen and touch input [4, 15] has been shown to significantly enhance pen interactions on a touch device [21, 40, 41]. Pfeuffer et al. [24] combined pen, touch, and gaze input for a target acquisition task. Pfeuffer et al. [25] further proposed thumb and pen interaction on tablets, including editing and formatting spreadsheets, web navigation and object manipulation. Romat et al. [29] explored data visualization metaphors using pen and touch as input.

These works demonstrate new possibilities when pen sensing is combined with touch. In contrast, our work primarily focuses on interactions that are possible when only the pen shaft is augmented for explicit interaction with a user interface.

2.2 Feedthrough Capacitance Sensing

Significant advances in capacitance sensing have enabled a host of novel devices and interactive capabilities [11]. We briefly present applications of feedthrough capacitance sensing, as used in PenShaft, which uses a layer of indirection to relay input from an inactive surface onto a capacitance sensing surface.

Rekimoto was the first to propose using feedthrough capacitance sensing to create *capacitance tags*, where portions of an object are covered with conductive material so that touch input can be redirected to an underlying adjoining capacitive display [27]. A drawback of this approach is that the tangible can only be detected when it is being touched [11]. Using passive sensing also greatly simplifies tangible design, as no additional components (such as a powered battery) beyond the tangible and the conductive material are required.

The typical method of instrumenting tangibles is to arrange conductive materials in unique configurations on the bottom of the object to create false "touch points" that are used as identifiers [6, 10, 39]. Touchscreen devices can use the identifier to activate specific functionalities and detect the location and rotation of an object. Researchers have instrumented everyday objects such as a knob [18], drawing tools [3], game pieces [5], and a smartphone [23]. Ikeda and Tsukada created a marker that can be recognized by a touchscreen and a camera [16].

We are not aware of any works that instrument a pen shaft for feedthrough capacitance sensing on an interactive surface to enable "buttons", "sliders", or other actions as we do.

3 PEN SHAFT CHARACTERISTICS

We describe the characteristics of PenShaft, beginning with basic pen shaft detection, and how basic detection can be extended for further interaction.

3.1 Basic Pen Shaft Detection

Capacitive touchscreens operate on the principle that when a human finger or body part comes close to the touchscreen, the body either becomes a capacitor that drains current from the screen surface, or absorbs the electromagnetic field [36]. Sensors in the touchscreen are able to detect touch events, and calculate the trajectory of a moving finger from repeated touch events.

Rekimoto [27] first proposed leveraging this principle to create *capacitance tags*, allowing tangible objects to be detected on a touch screen through the use of what Schmitz et al. term *forwarding conductors*, or a *forwarder* [30]. The *forwarder* is applied to the tangible so that it provides a path from a human finger to the touchscreen, thus "forwarding" the touch to the touchscreen.

To detect the pen shaft, we use a conductive material (copper tape) as a *forwarder*. We distinguish the pen from regular touch or multi-touch events by wrapping the material around the pen to create a minimum of three contact points. These contact points are connected with another strip of conductive material, so any touch along the conductive strip is forwarded to the touchscreen.

The pen is identified when three or more touch events appear on the touchscreen in a straight line. This pattern of touch events is distinguishable from regular touch interaction as it is difficult for users to place three or more fingers on a touchscreen in a perfectly straight line at the exact distances between the contact points.

3.2 Design Constraints

We place several constraints on the design of PenShaft devices. First, contact points cannot be placed too closely together. If contact points are not kept at a certain minimum distance apart, the touchscreen will not be able to distinguish each contact point as a separate touch event when the pen is placed on it. The specific minimum distance required between touch events varies for each device, but on our testing device, this distance is 1cm.

Second, there is a limit on the number of contact points that can be placed on a pen shaft, as current touchscreen devices only support a fixed number of simultaneous touch events. Our testing device is limited to ten simultaneous touch events.

The third constraint is the design of the pen itself. Through testing, we have found that pens with cylindrical shafts result in degraded tracking performance. Users find it difficult to grip a cylindrical shaft while maintaining good contact with the touchscreen. We thus recommend using a pen shaft with at least one flat surface, which facilitates consistent contact with the touchscreen. Our prototype uses a pen with a cuboid shaft.

3.3 On-Pen Interaction

We further extend basic pen shaft detection to create "buttons" on the pen shaft, greatly increasing PenShaft's interaction capabilities.

We propose the design in Figure 2a. We place conductive materials around the end of a pen shaft in three places to create the contact points P_1, P_2, and P_3, which are connected with additional conductive material. Since the distances between P_1 and P_2 (d_1) and P_2 and P_3 (d_2) are different, the pen's orientation can be determined. Note that the P_1 to P_3 area is not required to be the same size as in Figure 2. This area can be as small or as large as one wishes, so long as d_1 and d_2 are not equal. Buttons are then implemented by

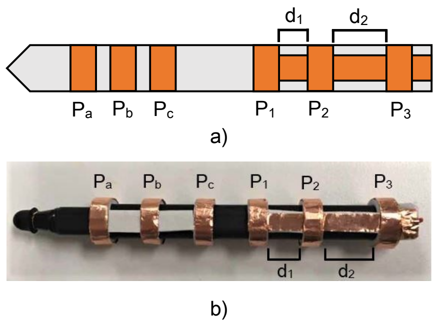


Figure 2: a) A diagram and b) a prototype of PenShaft device with buttons.

adding another three contact points to the front of the pen shaft, P_a, P_b, and P_c. These points are not connected to each other, or to the points P_1 to P_3. Figure 2b shows a prototype pen implementing this design. To add regular pen tip interaction to our PenShaft prototype, we attached a rubber tip to the tip of the pen and ran a conductive cable through the pen barrel from the pen tip to the P_1-P_3 area. This pen tip will only function if the user's palm is contacting the P_1-P_3 area.

When users place the pen shaft on a touchscreen while holding the P_1 to P_3 area, P_1, P_2, and P_3 are detected. The system recognizes the pen from seeing these three points in a straight line. We can also detect the direction the pen is facing using the asymmetric layout of d_1 and d_2. If the user places another finger on P_a, P_b, or P_c, a fourth contact point will be detected. We can determine that this contact is part of the pen because it extends the straight line formed by P_1, P_2, and P_3 Which of the three buttons was pressed can be distinguished based on the fourth contact point's distance from P_1, P_2, and P_3. Sliding gestures can also be detected when one button press occurs after another within a predefined time span.

The design shown in Figure 2 can be modified. The number of buttons can be smaller or larger than three, based on the amount of space available on the pen, and the minimum distance a device requires between touches.

3.4 Combining Pen Shaft and Touch Interaction

Various works have proposed combining pen and touch input from the non-dominant hand [15], taking advantage of the precision of the pen tip and finger dexterity. With PenShaft, we extend pen and touch interaction to a larger design space. When placing PenShaft on the screen, users can perform any of the above-mentioned interactions with one hand, and use the other hand to perform other tasks.

4 FUNDAMENTAL INTERACTIONS

To help design PenShaft interfaces, we define six fundamental PenShaft interactions. Interactions are divided into two categories: interactions that use the entire pen shaft (Shaft+Drag, Shaft+Rotate), pen interactions.jpg

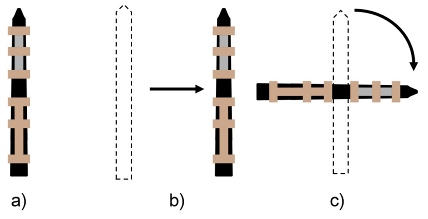


Figure 3: Whole-shaft interactions, including a) Stationary, b) Shaft + Drag, c) Shaft + Rotate

and interactions that use on-pen interaction, as defined in the previous section (Shaft Click, Shaft Click+Hold, Shaft Slide). In the following sections, interactions are described using the pen design in Figure 2.

4.1 Stationary

The system detects the pen when it is placed on or lifted off the touchscreen (Figure 3a). This interaction is the starting point for all other five interactions. These interactions can only be performed and detected after the pen has been placed on the touchscreen. Stationary can be used to split the screen into two views when the pen is placed on the screen.

4.2 Shaft + Drag

Once the pen is placed on the touchscreen, it can be reliably dragged to another location on the screen, then lifted (Figure 3b). Shaft + Drag can be used as a "brush" to select or erase objects.

4.3 Shaft + Rotate

Once the pen is placed on the touchscreen, it can be rotated (Figure 3c). Shaft + Rotate can be used to rotate content on the screen, such as a map or a picture, or could be used like a control knob.

4.4 Shaft Click

While the pen is placed on the touchscreen, the user presses and releases P_a, P_b, or P_c to "click" a button (Figure 4a). This interaction includes tapping on a contact point any number of times, or tapping on multiple contact points at once. Shaft Click allows for a great number of interactions, such as menu navigation.

4.5 Shaft Click + Hold

While the pen is placed on the touchscreen, the user presses P_a, P_b, or P_c but does not release their finger (Figure 4b). This interaction is an extension to Shaft Click, and can be considered equivalent to a "long press". The dwell time required to activate this interaction can be adjusted for different applications, or for a user's preferences. Shaft Click + Hold can be used for opening a menu with additional options, or increasing line width while a button is held. It can also be combined with Shaft + Drag to implement a form of rectangular select.

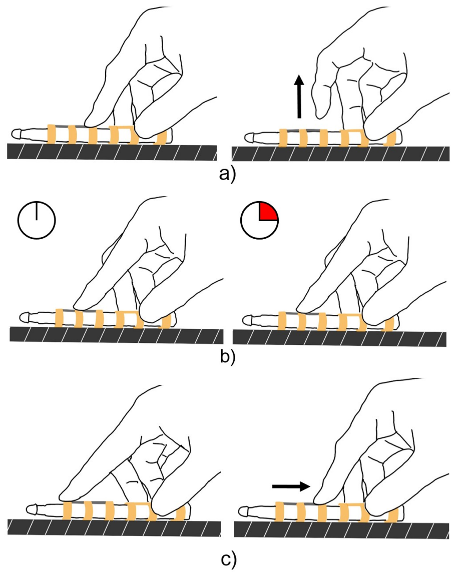


Figure 4: On-pen interactions, including a) Shaft Click, b) Shaft Click + Hold, c) Shaft Slide

4.6 Shaft Slide

While the pen is placed on the touchscreen, the user slides their finger from one contact point to another (Figure 4c). For example, the finger can be slid from P_a to P_b, P_a to P_c, P_b to P_c, P_c to P_a, P_b to P_a, and so on. Shaft Slide can be used for adjusting a slider (adjusting system volume, for example).

Theoretically, the maximum number of sliding gestures available on a pen is a 2-permutation of n with no repetitions, where n is the number of buttons on the pen. For our prototype, this means six sliding gestures are available, which offers a wide selection of slide actions to map to UI actions.

5 VALIDATION STUDY

We conducted an assessment of each PenShaft interaction. Our objectives were to validate the performance of our proposed PenShaft implementation and the system's ability to detect our interactions. We also evaluate the usability of the PenShaft interactions when performed by users and gather feedback about the interactions and PenShaft use in general.

5.1 Participants

6 participants (2 female, all right handed) volunteered from within our organization. Participants ranged in age from 23 to 36 (M=30.2, SD = 5.8).

5.2 Apparatus

We used a Samsung Galaxy Tab A 10.1 running Android 9.0. Each task was implemented as a custom Android application, and all touch events were logged.

The prototype pen used for the study (as seen in Figure 2) measured 116 mm in length and 8 mm in width. It was instrumented

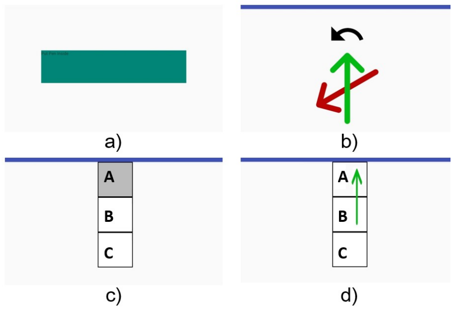


Figure 5: The experimental UIs for the a) Stationary, b) Rotate, c) Click and Click + Hold, and the d) Slide task

with the design in Figure 2b, and had three contact points spaced 16 mm apart for use as buttons.

5.3 Tasks

Each task evaluated a different PenShaft interaction.

Stationary — Participants placed the PenShaft device on randomly placed rectangles 30 times. These rectangles were 800px wide and 180px high. A trial was an error if the device was not fully within the rectangle. This task evaluated participants' ability to place the device on a touchscreen at random locations and orientations, and validated the system's ability to detect the device when placed on the touchscreen.

Shaft + Drag — A 1D Fitts' Law dragging task similar to [17] and [20] was evaluated with one target width (W = 120px) crossed with 4 distances (D = 400px, 800px, 1200px, 1600px). ID ranged from 2.12 to 3.46. We chose to not vary target width, as this task is meant to evaluate the dragging interaction, not target acquisition. Participants completed 10 trials for each distance.

Shaft + Rotate — Two arrows in green and red were displayed on the screen (Figure 5a). Participants rotated the green arrow to match the red arrow in a single motion. If the green arrow was not within 5 degrees of the red arrow's rotation at the end of a motion, the trial was considered an error, and participants had to redo it. Participants were shown feedback after every trial to indicate success or failure. Four angles were tested (30°, 60°, 90°, and 120°) in both clockwise and counterclockwise directions. Participants completed 10 trials in both directions for each angle. The order of rotation directions was randomized for each participant.

This task is similar to one used by Zhao et al. [42], though our task does not incorporate object translation and scaling.

Shaft Click — Participants were shown the screen in Figure 5b. When a button was highlighted, they tapped that button on the PenShaft device. Participants had to press each button (A, B or C) 10 times. The order of button presses was randomized for each participant. Participants were shown feedback after a button press to indicate success or failure.

Shaft Click + Hold — This task was identical to the *Shaft Click* task, but participants had to hold a button for 1 second. A counter was displayed in the top left corner of the screen. When this counter reached 1000 (milliseconds), a trial was complete. Again, participants had to press each button 10 times, with press order randomized.

Shaft Slide — Participants were shown the screen in Figure 5c, with an arrow indicating the start, end, and direction of a sliding gesture to complete. Feedback was shown to indicate whether a gesture was successfully completed. Participants performed each of the six sliding gestures (A-B, A-C, B-C, C-A, C-B, and B-A) 10 times, in random order.

5.4 Study Design

A within-subjects design was used. The primary independent variable was TASK, with six levels: STATIONARY, DRAG, ROTATE, CLICK, HOLD, and SLIDE. Participants completed 30 STATIONARY trials, 40 DRAG trials, 80 ROTATE trials, 30 trials each for CLICK, and HOLD, and 60 SLIDE trials, for a total of 270 trials per participant.

For all tasks, the primary measures were *Trial Time* and *Error Rate*. *Trial Time* is the time from the start of the trial, to trial completion. *Error Rate* is the percentage of trials where an error occurred. We collected data for the subjective measures *Fatigue*, *Comfort*, *Ease of Use*, *Perceived Speed*, and *Perceived Accuracy*. Subjective rankings used a scale from 1-7, where a higher score meant lower *Fatigue*, higher *Comfort*, better *Ease of Interaction*, and higher *Perceived Speed* and *Perceived Accuracy*. For ROTATE and SLIDE, we also asked participants to rank each rotation angle or slide gesture for each subjective measure. In these rankings, a lower rank was better.

5.5 Procedure

The experiment consisted of a single 40 minute session. Before beginning each task, the experimenter demonstrated the task to participants, who then completed a training block with 5 trials to familiarize themselves with the PenShaft device and the tasks. Participants completed the tasks in a fixed order.

After completing each task, participants completed a survey to gather subjective data. For the Rotate and Slide tasks, participants ranked each rotation angle and sliding gesture for each of the subjective measures.

6 RESULTS

For each TASK, trials with a *Trial Time* more than 3 standard deviations from the mean were considered outliers and excluded. In all tasks except STATIONARY, this exclusion was performed for each condition. Three outliers (1.67% of data) were removed for STATIONARY, 3 (1.25%) for DRAG, 6 (1.36%) for ROTATE, 6 (2.86%) for CLICK, 4 (1.9%) for HOLD, and 8 (2.22%) for SLIDE.

6.1 Stationary

The average time per trial was 1.00s, with an average error rate of 1.19%. Participants rated STATIONARY very favourably, with all subjective measures having an average rating above 6.

6.2 Shaft Drag

The average trial time was 1.57s. Participants were fastest when D = 400px (1.27s), and slowest when D = 1600px (1.93). All values of D had an *Error Rate* of 0%. Again, participants rated DRAG favourably, with average ratings above 5.5 for all subjective measures.

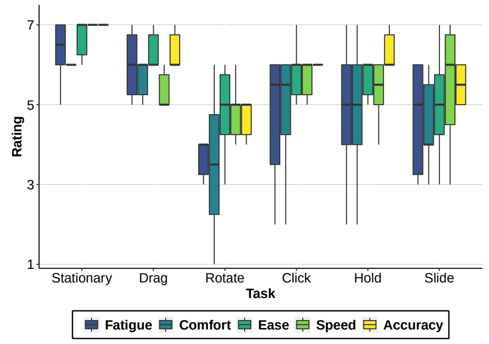
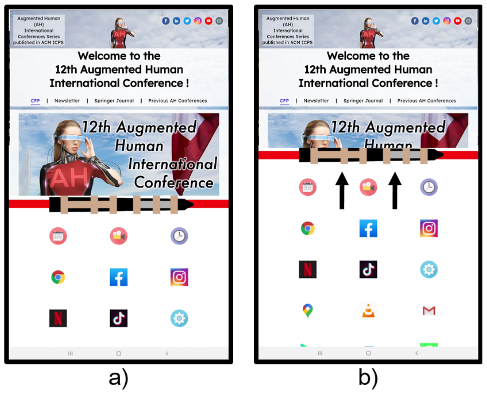


Figure 6: Average Task Ratings (a higher rating is better)



split.PNG

Figure 7: a) The screen is split into two views at the pen's location. b) The views are resized by dragging the pen.

6.3 Shaft Rotate

Each trial took 2.04s on average, with an overall *Error Rate* of 7.08%. Participants were fastest when rotating 30° (1.53s), and slowest when rotating 120° left or right (2.52s). -60° had the lowest *Error Rate* (1.67%), and 90° the highest (13.33%).

Participants rated ROTATE neutrally for *Ease of Interaction*, *Perceived Speed*, and *Perceived Accuracy*, but rated this gesture somewhat negatively for *Fatigue* and *Comfort*. -30°was ranked highly on average for all subjective measures, while 120°was ranked last for all subjective measures. In general, participants preferred counter-clockwise rotations, and disliked larger rotations.

6.4 Shaft Click

Overall, each trial took 1.96s, with an average *Error Rate* of 0%. Button C was the fastest (1.83s), and A was the slowest (2.06s). All buttons had an *Error Rate* of 0%. Participants rated CLICK somewhat favourably for all measures except *Fatigue* and *Comfort*, which were rated neutrally.

6.5 Shaft Click + Hold

Each trial took 2.58 secs (1.58s when subtracting hold time), with an average *Error Rate* of 1.94%. Button C was the fastest, and A was the slowest by 614ms. Buttons C and B had the lowest *Error Rates* (0.83%), and button A had the highest (4.17%). Participants rated HOLD favourably for all measures except *Fatigue* and *Comfort*, which were rated neutrally.

6.6 Shaft Slide

Each trial took 2.27s on average, with a 12.8% *Error Rate* overall. Gesture B-C was the fastest with an average *Trial Time* of 2.06s, and Gesture C-A was slowest (2.44s). B-C also had the lowest average *Error Rate* (3.33%), and B-A had the highest (20.19%).

Participants rated SLIDE somewhat favourably for all measures except *Fatigue* and *Comfort*, which were rated neutrally. B-C ranked high on all subjective measures, and C-A was ranked poorly for all measures. Gestures where the finger moved *down* the shaft ranked higher than gestures where the finger moved *up* the shaft.

7 APPLICATIONS

We designed several applications to demonstrate PenShaftś fundamental interactions, and how PenShaft can be used to augment pen interaction.

7.1 Screen Splitting

Users can split the screen into two views by placing the PenShaft device on the tablet screen, with the split appearing at the pen's location (Figure 7a). The split views can be resized by acquiring the dividing line with the pen, and dragging it back and forth (Figure 7b). When using the pen tip with a full screen application, e.g. a drawing app, a notes app is automatically opened on the other side of the screen by using PenShaft to split the screen.

7.2 Colouring Application

In a simple colouring application, PenShaft is used as a tangible eraser. When the device is dragged across the touchscreen, the selected color is erased. This interaction is similar to how artists use physical materials for painting.

7.3 2D Object Manipulation

PenShaft enables 2D object manipulation without requiring additional menus or widgets to change modes. In our example application, objects are selected by tapping on them with the pen tip. The selected object can then be rotated by rotating the PenShaft device (Figure 8a), scaled by swiping the finger up and down the shaft (Figure 8b), and translated by dragging the device. Selected objects can also be translated with the pen tip.

7.4 Enhanced Drawing Application

We enhance a standard drawing application with PenShaft. When the pen is placed on the touchscreen, a menu appears next to the pen, allowing users to change the pen's mode, its attributes (similar to Zhang et al. [41]), and global options. The menus are navigated by clicking on the "buttons". The user can return to the top-level menu by swiping from the top button to the bottom button. By moving these menus to the PenShaft, the user can modify the pen's

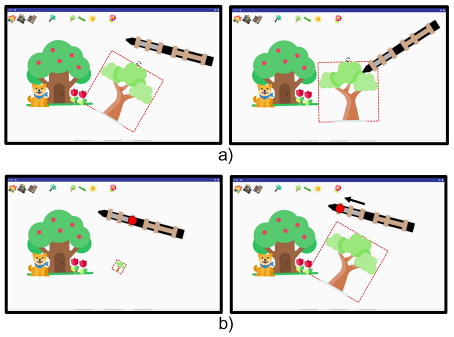


Figure 8: A selected object can be a) rotated, by rotating the PenShaft, and b) scaled, by swiping the finger up the pen shaft.

mode and attributes without interacting with GUI elements located elsewhere on the screen.

The pen can switch between three modes: Brush, Erase, and Draw. In the Brush and Erase Modes, users drag the pen to either color large portions of the screen, or erase drawings. Brush and Erase modes remain active while the user presses that respective button. Once the button is released, the menu reappears.

Users can modify the color, thickness, and transparency of the pen's stroke. Color is modified by individually selecting the red, green, or blue components, and rotating the pen. A red, green, or blue value of 0 is mapped to 270°, and a value of 255 is mapped to 90°. Users can view color changes in a square at the top of the screen. Thickness is modified by selecting one of three options; small, normal, and large. Transparency is modified by swiping the finger up or down the pen shaft.

Finally, users can modify the device's brightness by dragging the pen shaft horizontally, and the device's volume by tapping "Volume Up" and "Volume Down" buttons on the pen shaft. There is also an option to toggle which side of the pen the menu appears on, to accommodate left-handed and right-handed users. Please refer to the supporting video for more details about this application.

7.4.1 Eyes Free Expert Mode. The drawing application also has an eyes-free mode, where menu options are accessed by quickly tapping the buttons, similar to a shortcut key. For instance, one could access the color modifier by tapping button B, then button A in rapid succession.

8 DISCUSSION

8.1 Validation Results

All tasks but two had an overall error rate of less than 5%. The elevated error rates in the rotation task are not overly surprising, given that participants found larger rotations of 90°or more to be fatiguing and uncomfortable. When performing these larger rotations, the starting position of the hand and wrist becomes important, as the user may be forced to contort the hand and wrist into uncomfortable positions to complete the rotation. Since all our participants were right-handed, it is also natural that they would find counterclockwise rotations to be more comfortable.

As for the sliding task, it is evident that some sliding gestures are more difficult to complete than others. This is not overly concerning, as implementing all possible sliding gestures in an interface is impractical. The least error-prone gestures, such as B-C and A-B would be best suited for applications.

8.2 Design Considerations and Limitations

While observing participants in the validation study, several design considerations for future PenShaft applications emerged. Using PenShaft for extended periods is fatiguing. Participants noted that their hands and wrists became increasingly sore after using the the PenShaft buttons for the clicking, clicking and holding, and sliding tasks. Applications that use PenShaft should keep aim to keep PenShaft interaction short.

The rotation task results indicate that applications should refrain from requiring users to make large rotations with the PenShaft device, as users do not find these rotations comfortable. The sliding task results also indicate that sliding *down* the pen shaft is preferable to sliding *up*.

We also observed that the pen blocks content when placed on the touchscreen, an issue we term "the fat pen problem". Developers must take this into consideration when designing PenShaft interfaces, though the fat pen problem could be mitigated to some extent with eyes-free interaction.

The greatest limitation of the study is that we were unable to recruit a large number of participants due to the COVID-19 pandemic.

8.3 Future Work

In future iterations, many aspects of PenShaft could be improved. For instance, it is possible to differentiate between each pen side by varying the number of contact points on each side, and the distance between them. This would enable similar functionality as Conté [8, 35], where different functions are mapped to different sides of a device. If the pen were actively powered, many more augmentations could be made to better track the pen shaft, or enable 6DOF interaction.

It is important to examine the ergonomics of PenShaft. In the validation study, one participant found it more comfortable to operate the PenShaft device with two hands, instead of one. Several participants also noted clicking and sliding on the pen shaft with one hand for prolonged periods was tiring. It is important to examine which grips are most comfortable and efficient for users. The spacing between buttons is also important for PenShaft ergonomics, as it determines how far users need to stretch their fingers to operate the buttons.

The transition cost for switching between a typical pen grip and PenShaft is also critical. While a two-handed grip may be more comfortable, it may take more time to transition to and from this grip. These costs are critical if PenShaft were to be used in real world applications. Strategies for switching between normal pen use and PenShaft should be examined to determine what is more efficient and ergonomic for users.

9 CONCLUSION

PenShaft is a battery free augmentation for passive capacitive pens. Users can perform *whole-shaft* and *on-pen* interactions with PenShaft, enabling a wide variety of applications. Our validation study found all but two PenShaft interactions have a recognition rate over 95%. We demonstrate that our augmentation to a capacitive pen can support a wide range of pen-based interactions, including the ability to engage with hotkeys, menus, and image manipulations, directly via the pen shaft.

PenShaft is a flexible medium through which application designers can directly embed novel interactive capabilities using a pen, with many future avenues for investigating PenShaft interaction, and for improving the device itself.

REFERENCES

[1] Apple. 2020. Apple Pencil. https://www.apple.com/apple-pencil/

[2] Xiaojun Bi, Tomer Moscovich, Gonzalo Ramos, Ravin Balakrishnan, and Ken Hinckley. 2008. An exploration of pen rolling for pen-based interaction. In *Proceedings of the 21st annual ACM symposium on User interface software and technology.* 191–200.

[3] Rachel Blagojevic and Beryl Plimmer. 2013. CapTUI: geometric drawing with tangibles on a capacitive multi-touch display. In *IFIP Conference on Human-Computer Interaction.* Springer, 511–528.

[4] Peter Brandl, Clifton Forlines, Daniel Wigdor, Michael Haller, and Chia Shen. 2008. Combining and measuring the benefits of bimanual pen and direct-touch interaction on horizontal interfaces. In *Proceedings of the working conference on Advanced visual interfaces.* 154–161.

[5] Dan Burnett, Paul Coulton, and Adam Lewis. 2012. Providing both physical and perceived affordances using physical games pieces on touch based tablets. In *Proceedings of the 8th Australasian Conference on Interactive Entertainment: Playing the System.* 1–7.

[6] Liwei Chan, Stefanie Müller, Anne Roudaut, and Patrick Baudisch. 2012. CapStones and ZebraWidgets: sensing stacks of building blocks, dials and sliders on capacitive touch screens. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems.* 2189–2192.

[7] Youngjun Cho, Andrea Bianchi, Nicolai Marquardt, and Nadia Bianchi-Berthouze. 2016. RealPen: Providing realism in handwriting tasks on touch surfaces using auditory-tactile feedback. In *Proceedings of the 29th Annual Symposium on User Interface Software and Technology.* 195–205.

[8] Lisa A Elkin, Jean-Baptiste Beau, Géry Casiez, and Daniel Vogel. 2020. Manipulation, learning, and recall with tangible pen-like input. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems.* 1–12.

[9] Nicholas Fellion, Thomas Pietrzak, and Audrey Girouard. 2017. FlexStylus: Leveraging bend input for pen interaction. In *Proceedings of the 30th Annual ACM Symposium on User Interface Software and Technology.* 375–385.

[10] Timo Götzelmann and Daniel Schneider. 2016. CapCodes: Capacitive 3D printable identification and on-screen tracking for tangible interaction. In *Proceedings of the 9th Nordic Conference on Human-Computer Interaction.* 1–4.

[11] Tobias Grosse-Puppendahl, Christian Holz, Gabe Cohn, Raphael Wimmer, Oskar Bechtold, Steve Hodges, Matthew S Reynolds, and Joshua R Smith. 2017. Finding common ground: A survey of capacitive sensing in human-computer interaction. In *Proceedings of the 2017 CHI conference on human factors in computing systems.* 3293–3315.

[12] Tovi Grossman, Ken Hinckley, Patrick Baudisch, Maneesh Agrawala, and Ravin Balakrishnan. 2006. Hover widgets: using the tracking state to extend the capabilities of pen-operated devices. In *Proceedings of the SIGCHI conference on Human Factors in computing systems.* 861–870.

[13] Khalad Hasan, Xing-Dong Yang, Andrea Bunt, and Pourang Irani. 2012. A-coord input: coordinating auxiliary input streams for augmenting contextual pen-based interactions. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems.* 805–814.

[14] Ken Hinckley, Michel Pahud, Hrvoje Benko, Pourang Irani, François Guimbretière, Marcel Gavriliu, Xiang'Anthony' Chen, Fabrice Matulic, William Buxton, and Andrew Wilson. 2014. Sensing techniques for tablet+ stylus interaction. In *Proceedings of the 27th annual ACM symposium on User interface software and technology.* 605–614.

[15] Ken Hinckley, Koji Yatani, Michel Pahud, Nicole Coddington, Jenny Rodenhouse, Andy Wilson, Hrvoje Benko, and Bill Buxton. 2010. Pen+ touch= new tools. In *Proceedings of the 23nd annual ACM symposium on User interface software and technology.* 27–36.

[16] Kohei Ikeda and Koji Tsukada. 2015. CapacitiveMarker: novel interaction method using visual marker integrated with conductive pattern. In *Proceedings of the 6th Augmented Human International Conference.* 225–226.

[17] Nikhita Joshi and Daniel Vogel. 2019. An Evaluation of Touch Input at the Edge of a Table. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems.* 1–12.

[18] Sven Kratz, Tilo Westermann, Michael Rohs, and Georg Essl. 2011. CapWidgets: tangile widgets versus multi-touch controls on mobile devices. In *CHI'11 Extended Abstracts on Human Factors in Computing Systems.* 1351–1356.

[19] Ernst Kruijff, Saugata Biswas, Christina Trepkowski, Jens Maiero, George Ghinea, and Wolfgang Stuerzlinger. 2019. Multilayer haptic feedback for pen-based tablet interaction. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems.* 1–14.

[20] I Scott MacKenzie, Abigail Sellen, and William AS Buxton. 1991. A comparison of input devices in element pointing and dragging tasks. In *Proceedings of the SIGCHI conference on Human factors in computing systems.* 161–166.

[21] Fabrice Matulic and Moira C Norrie. 2013. Pen and touch gestural environment for document editing on interactive tabletops. In *Proceedings of the 2013 ACM international conference on Interactive tabletops and surfaces.* 41–50.

[22] Microsoft. 2020. Surface Pen. https://www.microsoft.com/en-us/p/surface-pen/92FP8Q09QHXC?ICID=SurfaceAccCat_Hero1_SurfacePen_010120

[23] Gary Perelman, Marcos Serrano, Christophe Bortolaso, Celia Picard, Mustapha Derras, and Emmanuel Dubois. 2019. Combining tablets with smartphones for data analytics. In *IFIP Conference on Human-Computer Interaction.* Springer, 439–460.

[24] Ken Pfeuffer, Jason Alexander, and Hans Gellersen. 2016. Partially-indirect bimanual input with gaze, pen, and touch for pan, zoom, and ink interaction. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems.* 2845–2856.

[25] Ken Pfeuffer, Ken Hinckley, Michel Pahud, and Bill Buxton. 2017. Thumb+ Pen Interaction on Tablets.. In *CHI.* 3254–3266.

[26] Gonzalo Ramos, Matthew Boulos, and Ravin Balakrishnan. 2004. Pressure widgets. In *Proceedings of the SIGCHI conference on Human factors in computing systems.* 487–494.

[27] Jun Rekimoto. 2002. SmartSkin: an infrastructure for freehand manipulation on interactive surfaces. In *Proceedings of the SIGCHI conference on Human factors in computing systems.* 113–120.

[28] Jun Rekimoto and Eduardo Sciammarella. 2000. Toolstone: effective use of the physical manipulation vocabularies of input devices. In *Proceedings of the 13th annual ACM symposium on User interface software and technology.* 109–117.

[29] Hugo Romat, Nathalie Henry Riche, Ken Hinckley, Bongshin Lee, Caroline Appert, Emmanuel Pietriga, and Christopher Collins. 2019. ActiveInk: (Th) Inking with Data. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems.* 1–13.

[30] Martin Schmitz, Jürgen Steimle, Jochen Huber, Niloofar Dezfuli, and Max Mühlhäuser. 2017. Flexibles: deformation-aware 3D-printed tangibles for capacitive touchscreens. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems.* 1001–1014.

[31] Hyunyoung Song, Hrvoje Benko, Francois Guimbretiere, Shahram Izadi, Xiang Cao, and Ken Hinckley. 2011. Grips and gestures on a multi-touch pen. In *Proceedings of the SIGCHI conference on Human Factors in computing systems.* 1323–1332.

[32] Minghui Sun, Xiang Cao, Hyunyoung Song, Shahram Izadi, Hrvoje Benko, Francois Guimbretiere, Xiangshi Ren, and Ken Hinckley. 2011. Enhancing naturalness of pen-and-tablet drawing through context sensing. In *Proceedings of the ACM International Conference on Interactive Tabletops and Surfaces.* 83–86.

[33] Yu Suzuki, Kazuo Misue, and Jiro Tanaka. 2009. Interaction technique for a pen-based interface using finger motions. In *International Conference on Human-Computer Interaction.* Springer, 503–512.

[34] Daniel Vogel and Ravin Balakrishnan. 2010. Direct pen interaction with a conventional graphical user interface. *Human–Computer Interaction* 25, 4 (2010), 324–388.

[35] Daniel Vogel and Géry Casiez. 2011. Conté: multimodal input inspired by an artist's crayon. In *Proceedings of the 24th annual ACM symposium on User interface software and technology.* 357–366.

[36] Geoff Walker. 2014. Part 1: Fundamentals of Projected-Capacitive Touch Technology. *Intel Corporation, SID Display Week* 14 (2014), v1.

[37] Yizhong Xin, Xiaojun Bi, and Xiangshi Ren. 2011. Acquiring and pointing: an empirical study of pen-tilt-based interaction. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems.* 849–858.

[38] Xingya Xu, Jinxi Liao, and Hirohito Shibata. 2017. Drawing in talking: Using pen and voice for drawing system configuration figures in talking. In *Proceedings of the 25th International Display Workshops (IDW'17). Fukuoka, Japan.*

[39] Neng-Hao Yu, Li-Wei Chan, Seng Yong Lau, Sung-Sheng Tsai, I-Chun Hsiao, Dian-Je Tsai, Fang-I Hsiao, Lung-Pan Cheng, Mike Chen, Polly Huang, et al. 2011. TUIC: enabling tangible interaction on capacitive multi-touch displays. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems.* 2995–3004.

[40] Robert Zeleznik, Andrew Bragdon, Ferdi Adeputra, and Hsu-Sheng Ko. 2010. Hands-on math: a page-based multi-touch and pen desktop for technical work

and problem solving. In *Proceedings of the 23nd annual ACM symposium on User interface software and technology*. 17–26.

[41] Yang Zhang, Michel Pahud, Christian Holz, Haijun Xia, Gierad Laput, Michael McGuffin, Xiao Tu, Andrew Mittereder, Fei Su, William Buxton, et al. 2019. Sensing posture-aware pen+ touch interaction on tablets. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*. 1–14.

[42] Jian Zhao, R William Soukoreff, and Ravin Balakrishnan. 2015. Exploring and modeling unimanual object manipulation on multi-touch displays. *International Journal of Human-Computer Studies* 78 (2015), 68–80.