Investigating Performance and Usage of Input Methods for Soft Keyboard Hotkeys

Katherine Fennedy
Singapore University of Technology and Design (SUTD)
katherine_fennedy@mymail.sutd.edu.sg

Sylvain Malacria
Inria, Univ. Lille, UMR 9189 - CRIStAL, Lille, France
sylvain.malacria@inria.fr

Hyowon Lee
Dublin City University
hyowon.lee@dcu.ie

Simon T. Perrault
Singapore University of Technology and Design (SUTD)
perrault.simon@gmail.com

ABSTRACT

Touch-based devices, despite their mainstream availability, do not support a unified and efficient command selection mechanism, available on every platform and application. We advocate that hotkeys, conventionally used as a shortcut mechanism on desktop computers, could be generalized as a command selection mechanism for touch-based devices, even for keyboard-less applications. In this paper, we investigate the performance and usage of soft keyboard shortcuts or hotkeys (abbreviated *SoftCuts*) through two studies comparing different input methods across sitting, standing and walking conditions. Our results suggest that SoftCuts not only are appreciated by participants but also support rapid command selection with different devices and hand configurations. We also did not find evidence that walking deters their performance when using the Once input method.

CCS CONCEPTS

• **Human-centered computing** → **Interaction techniques**.

KEYWORDS

Hotkey, shortcut, soft keyboard, modifier-based shortcuts, command selection

ACM Reference Format:
Katherine Fennedy, Sylvain Malacria, Hyowon Lee, and Simon T. Perrault. 2020. Investigating Performance and Usage of Input Methods for Soft Keyboard Hotkeys. In *22nd International Conference on Human-Computer Interaction with Mobile Devices and Services (MobileHCI '20), October 5–8, 2020, Oldenburg, Germany*. ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3379503.3403552

1 INTRODUCTION

Hotkeys (also known as keyboard shortcuts) are a well-established shortcut mechanism available on every desktop keyboards that allows users to select most frequent commands rapidly [33]. Touch-based devices, on the other hand, do not offer such a unique and unified shortcut mechanism, available on every platform and application. This may be one reason why, despite the mainstream availability, touch-based devices still cannot compete with desktop operating systems (OS) for many productivity tasks [24].

Physical keyboards became an exception rather than the norm on tablets and smartphones, where they are usually replaced with soft keyboards (software keyboards) that are made available only when text entry is required. This may be why hotkeys never seem to have been considered as a viable command selection standard for these devices. Another possible reason is that the primary strategies used to convey hotkeys in the desktop OS are tooltips (when the cursor hovers a widget) and labels (next to the menu item in the menu bar), methods that are not readily adaptable to the touch-based OS.

Attempts to instantiate them can be found on some commercial touch-based OS like the latest Microsoft Surface [40] and Samsung [43] tablets. A subset of available commands can be selected using keyboard shortcuts on the soft keyboard, by holding/tapping a modifier (Control/Command) key and subsequently hitting the key associated with a command. Another, yet different, example could be found in previous implementations of the Swype keyboard [38] where gestures could be performed by swiping from a dedicated button to specific keys to activate commands. However, commands associated with these keys are not exposed to the user who has either to guess them, discover them through repeated tries, or through online resources like websites. Regardless, these mechanisms are limited to scenarios where the keyboard is displayed and constrained to one specific input method (taps or swipes). More surprisingly, these commercial solutions do not leverage the dynamic nature of soft keyboards' layout that could be changed at will, making the discovery and learning of hotkeys even easier than on desktop systems. In summary, the motivation to further study hotkey-based command selection is driven by the following reasons:

- Hotkeys on touchscreens already exist, albeit not standardized in terms of input method and visual presentation.
- It has the potential to become a multi-device command mechanism, available on desktops and touch devices, where advanced users can reuse pre-existing knowledge from using desktop computers or the other way around, which become especially relevant when using productivity applications across different platforms [1].
- The keyboard layout provides a familiar canvas to maintain spatial consistency across applications for a given command.

MobileHCI '20, October 5–8, 2020, Oldenburg, Germany

ACM ISBN 978-1-4503-7516-0/20/10…$15.00
https://doi.org/10.1145/3379503.3403552

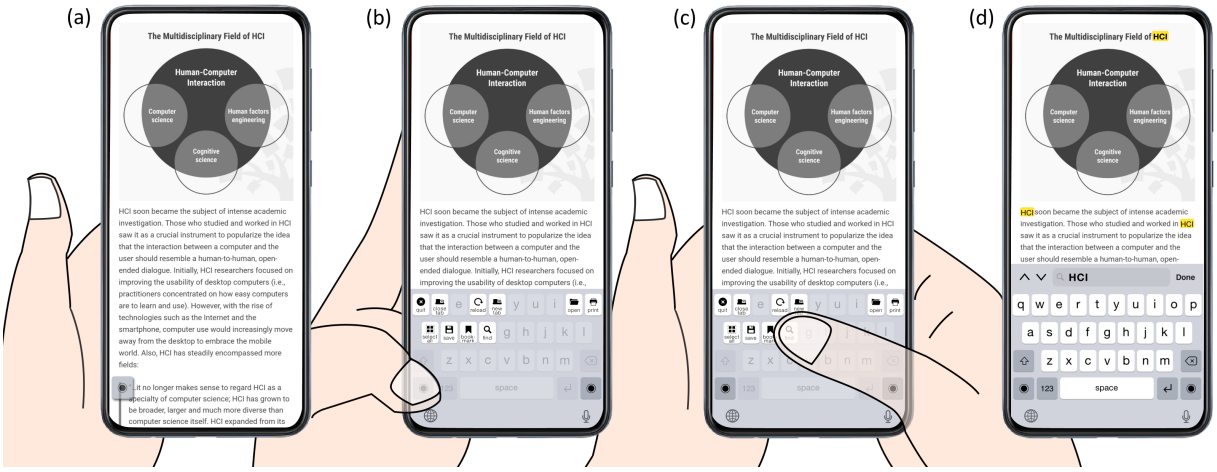

Figure 1: Web browsing scenario of soft keyboard shortcuts using *Once* method. (a) a semi-transparent modifier key is available in the browser, (b) when user taps it, all available shortcuts will be presented to facilitate browsing, (c) user then taps on the "F" key (find command) and hence, (d) every occurrence of the "HCI" word (as typed by the user) is highlighted in yellow.

In this paper, we advocate for the generalization of soft keyboard shortcuts or hotkeys (abbreviated *SoftCuts*) as a viable command selection mechanism on touch-based devices, and focus on investigating their performance and usage through two studies. Our first study compares the performance of three different input methods (*Once*, *Swipe* and *User Maintained*) for SoftCuts across different devices, orientations and number of hands used to interact regardless of the visual presentation. Our second study evaluates the usage of the same three input methods across varied mobility conditions like sitting, standing and walking. Our results suggest that while the input method *Once* (based on two successive taps on soft keys) overall performed best, some users also like using *Swipe* for one-handed interaction as well as *User Maintained* for Tablet. Altogether, these results confirm that SoftCuts could be generalized as an efficient command selection mechanism for touch-based devices.

2 RELATED WORK

2.1 Soft Keyboard Hotkeys in HCI

Currently, soft keyboard hotkeys have been successfully implemented by Samsung and Microsoft. We discuss these implementations in the next section. However, to the best of our knowledge, no academic work has been published on the topic. With a vast design space at our disposal, we decided to focus on input performance first, as it is one of the most important criteria considered in command selection literature.

2.2 Command Selection with Keyboard Shortcuts

Command selection on desktop computers mostly relies on pointing with the cursor on commands organized in menubars and toolbars. However, alternative methods using keyboards have been proposed. This is typically the case of Alt key navigation (part of the accessibility interface in Microsoft Office) that displays keys associated with successive levels of the menu hierarchy when the 'Alt' key is pressed, allowing users to select commands using the keyboard. However, it has been shown that users had difficulties in chunking these key sequences into one single cognitive unit [35]. Another alternative is keyboard shortcuts (also called hotkeys) that allow a wide range of commands to be selected with a single key combination, thus removing the need to traverse the hierarchy. Repeated studies have demonstrated the performance benefits of hotkeys over alternative input methods such as menubars or toolbars [11, 31, 35, 39]. The high performance ceiling of hotkeys has motivated researchers to investigate how they could leverage different keyboard capabilities [8], hand postures [48] or different types of key sequences [5]. Nevertheless, hotkeys still suffer from one main limitation, which is that users must *recall* the key combination in order to be able to use them [19, 34]. As a result, despite their performance benefits, hotkeys remain limited in use [10, 31]. Hardware solutions can be used to overcome the recall necessity of hotkeys. Typically, Optimus keyboards [45] are physical keyboards whose each key is a display that can dynamically change, typically to reveal the commands associated when a modifier key is pressed, hence making interaction with hotkeys rely on *recognition* rather than *recall*. Similar software solutions have also been designed so that users can expose the hotkeys on the monitor while pressing a modifier key [17, 33, 46] and demonstrated the efficiency of hotkeys even when users are not aware of the key combination beforehand.

2.3 Command Selection with Touch-based Devices

Similar to desktop computers and despite of supporting multitouch input capabilities, command selection on commercial touch-based devices still mostly rely on single point tap-based interaction, hence requiring selection commands to be located in toolbars or hamburger menus[1]. Exceptions will be found with *Swhidgets* buttons, hidden by default and that users first uncover with a swipe gesture

[1]https://en.wikipedia.org/wiki/Hamburger_button

relying on the metaphor of sliding physical objects [41, 44], or in device shaking gestures such as the shake-to-undo in iOS. Researchers though have long explored how different modalities, such as gestural interaction, could be adapted to facilitate command selection on touch-based devices. The widely studied marking menus [28] have influenced many stroke-based strategies [4, 14, 15, 25, 32, 47] to trigger commands on multitouch platforms. Similarly, MelodicTap [22] and Arpège [16] investigated rhythmic chorded gestures to expand input vocabulary on touch screens. However, as Norman pointed out, gestural interaction can be "a step backward in usability" [37] for reasons like poor discoverability and inconsistent mapping. The unnatural [36] and invisible mapping between a gestural input and the corresponding command output is why users need to learn and rehearse before being able to invoke such shortcuts on-demand [6]. FastTap [20] and its adaptation to smartwatches [30] alleviate this problem by relying on a full-screen static grid of commands that can be triggered through a chorded selection. Compared to SoftCuts, FastTap relies on a full-screen grid layout that supports user-maintained (UM) chording interaction only and is rehearsal-based. It offers both novice and expert modes of interaction, where for the latter, the menu only gets displayed after a delay. However, several studies have suggested that users have difficulties to adopt the expert mode [18, 29] and a recent study also investigated the necessity of the delay-based appearance [21] of Marking Menus, hence raising similar questions for existing techniques, such as FastTap.

Command selection techniques leveraging soft keyboard have also been explored in the literature. Initially designed for "pen-based" devices, one early example can be found in Command Strokes [27] where users can select a command by drawing its name on a soft keyboard, starting from the Ctrl key, as they would with a gestural keyboard [26]. More recently, Alvina et al. presented CommandBoard [2], which combines gestural text entry with an Octopocus menu [9] to merge typing and command execution into one continuous user action. Novice users can pause after having written the word to display the gestural menu and select the desired command. Ando et al. [3] proposed a different technique merging text selection with command execution, by starting a drag gesture from a specific key of the keyboard. The first key specifies the command while the drag gesture is used to select the desired text. However, this technique is limited to text associated commands and does not teach which command is associated with each key.

As a summary, while various command selection mechanisms for touch-based devices have been proposed, some leveraging the keyboard area, the performance and usage of hotkeys on soft keyboards have not been investigated yet. SoftCuts would have the advantage of being easy to discover (with an additional key on the soft keyboard layout), and we could leverage users' familiarity with that mechanism as it is already widely implemented on desktop and laptop computers using physical keyboards.

3 SOFTCUTS

As seen in previous sections, it is surprisingly rare for soft keyboards to support the activation of hotkeys. However, we believe that soft keyboard shortcuts/hotkeys (*SoftCuts*) could easily be generalized for every touch-based device, not only as a shortcut mechanism but as a generic command selection mechanism. Since soft keyboards are displayed on a screen, they are dynamic in nature and can adapt their layout and appearance. For example, the existing soft keyboard layout would change to display capital letters when the "Shift" key is pressed. It could similarly be adapted [13] when a modifier key (e.g. Ctrl or Cmd) is pressed to display its associated command on each key, therefore helping users to discover the commands.

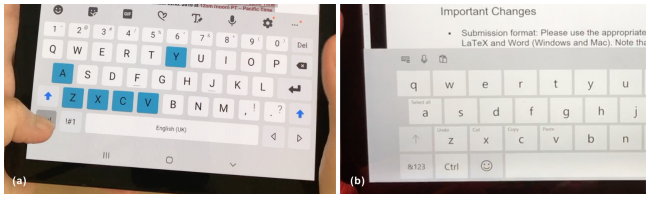

Figure 2: Existing Implementations of Soft Keyboard Hotkeys on (a) Samsung Galaxy S, with the hotkeys highlighted in blue once the user presses the Ctrl key, (b) Microsoft Surface, with the name of the commands associated with hotkeys displayed after the user taps on the Ctrl key. Note: Samsung uses User Maintained as an input method, while Microsoft uses Once.

3.1 Existing Implementations

Latest Microsoft Surface [40] and Samsung Galaxy S [43] tablets support the selection of a subset of available commands using keyboard shortcuts (Figure 2), by respectively hitting or holding a modifier key (Ctrl) and then hitting a key associated with a command. However, commands are poorly suggested to the user: in Microsoft implementations, the command is scarcely displayed using only text over the associated key, where on Samsung tablets, keys associated with commands are only highlighted without describing which command would be selected. As a result, these commercial solutions do not fully leverage the dynamic nature of soft keyboards that may make the discovery and learning of hotkeys even easier than on desktop systems, as the keyboard layout could be changed at will. Moreover, these mechanisms are limited to scenarios where the keyboard is displayed and constrained to one specific input method.

3.2 Input Methods

Unlike desktop hotkeys, SoftCuts do not require users to memorize the hotkeys beforehand, but leverage spatial memory, making them a viable command selection technique. SoftCuts also allow three different input methods to coexist:

(1) *User Maintained interaction (UM)*: Holding the modifier key with one finger and activating the hotkey with a different finger. This mechanism is similar to hotkeys on desktop computers and is also used on Samsung Galaxy S tablets.
(2) *Two sequential taps (Once)*: Tapping the modifier key (without holding) changes the layout of the keyboard to display commands that can then be selected. Subsequent tapping on

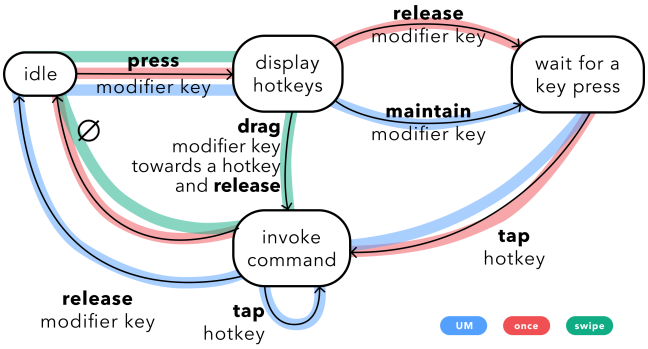

Figure 3: State diagrams for each *Input Method*, showing that the methods can be used concurrently on a system. Note: *UM* allows user to perform multiple command selections by tapping multiple hotkeys sequentially.

any key will change the layout back to QWERTY automatically (see Figure 1). This mechanism is similar to how the Shift key behaves to capitalize text with soft keyboards, and to how hotkeys are implemented in recent Microsoft Surface tablets.

(3) *Sliding gestures (Swipe)*: Touching the modifier changes the layout of the keyboard to display the commands, that can be selected by sliding the finger over the key and releasing it. This mechanism is similar to the currently discontinued Swype keyboard's implementation.

We illustrate how these three input methods are compatible with each other, in the state diagram (see Figure 3), leaving users the freedom to choose their preferred method depending on the context.

In the following sections, we will elaborate on two studies investigating the performance and usage of each input method. In the first experiment, we investigate the performance of each method and see which one(s) the user prefers. While a previous study [12] theoretically estimated that a swipe is approximately 10ms faster than a single pointing gesture, it is unclear how that result would translate to our unique context of keyboard layout and two sequential taps instead of a single one. In the second experiment, we investigate how different level of physical activity may impact the usage frequency. Which method performs the best? Do people switch methods as they wish, or is there a particular one they would prefer? How will the results change in different mobility conditions? The answers will be revealed next.

4 EXPERIMENT 1: PERFORMANCE OF INPUT METHODS

In this study, we compare the performance of the three SoftCuts' input methods (*UM*, *Once* and *Swipe*) in terms of time, accuracy and subjective preference. Since we envision SoftCuts as a general command selection mechanism for touch-based devices, we decided to test these input methods in various configurations, with both mobile phones and tablets, in landscape and portrait orientations and with one and two hands to perform the interaction.

4.1 Participants and Apparatus

Twelve participants (all right-handed, 5 female), aged 18 to 30 (M = 25.1, SD = 3.29) were recruited from the university community. They received the equivalent of 7.5 USD for their participation. One participant used a tablet every day while three used it at least once per month. Only one participant was familiar with swipe-based keyboards. The experimental software was written in Java using Android Studio as an IDE and ran on version 28 of Android SDK. We used a Samsung A10 phone (6.2", 168 grams) running Android 9 for the Phone condition and a Samsung Galaxy Tab 4 (10.5" screen, 483 grams) running Android 9 for the Tablet condition.

4.2 Procedure, Task and Stimulus

Participants began the experiment with a questionnaire that asked about their demographic information, familiarity with tablet and use of swipe-based keyboards. They were then briefed on the experiment and were presented the three *Input Methods* by the experimenter. They would then be asked to perform repeated series of command executions as rapidly and accurately as possible, with a given *Input Method* and for a given *Configuration*. A *Configuration* corresponds to the combination of a *Device* (Phone or Tablet), a given *Orientation* (Portrait or Landscape) and *Handedness* (one or two hands). The *Configurations* compared in this study are detailed below.

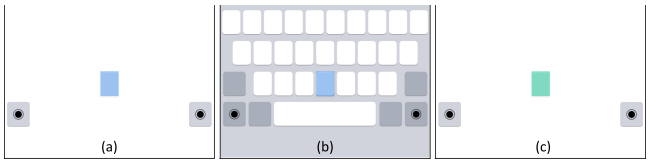

Figure 4: Example of a trial on smartphone. (a) The stimulus is highlighted in blue at the beginning of the trial to minimize visual search (b) the user pressed on one of the modifier keys, (c) upon correct selection, the key would turn green (red if incorrect).

As we were interested in the performance of command execution and wanted to eliminate other factors such as visual search, each trial began by specifying the target command by simply highlighting its key in blue (Figure 4(a)), which is common in command selection experiments [7]. Participants then had to select the target command using the requested *Input Method*. In that respect, participants could select the commands using either of the two modifier keys, respectively located on the bottom left and right corners of the keyboard (Figure 4). After performing the selection, feedback would appear on the screen for 500 milliseconds with the target being highlighted in either green (if correct) or red (if incorrect), before proceeding to the next trial (Figure 4(c)). A trial was considered successful if the participant was able to select the correct target. Trial completion time was measured from target appearance to target selection.

There were 6 possible targets (Q, Z, T, V, P and M) as these keys are spread around the keyboard. For each condition (*Device*×*Orientation*×*Handedness* combination), participants repeated

1 training block of 6 trials (once each target, in random order), followed by 3 test blocks of 12 trials (twice each target, in random order). After each condition, participants would provide subjective feedback on a 1-5 Likert Scale (1: strongly disagree, 5: strongly agree) on their overall opinion for the method ("I liked using this input method"), ease of use, comfort, perceived speed and perceived accuracy. Participants then proceeded to the next condition for that part.

4.3 Design

4.3.1 Experimental Conditions. Our initial design space for this experiment includes: Input Methods {UM, Once, Swipe}, Device {Phone, Tablet}, Handedness {One, Two}, Orientation {Landscape, Portrait}. Complete factorial exploration of this design space would lead to a total of 3 × 2 × 2 × 2 = 24 combinations. However, we decided to discard 10 combinations that we considered as impractical or impossible, to focus our study on 14 combinations that were considered comfortable and/or doable (see Table 1). The experiment was split in two parts (one- and two-handed). The order of appearance of both parts was fully counter balanced across participants.

Table 1: Experimental Conditions for Study 1. Greyed-out conditions were not tested: UM could not be tested in 1-handed conditions as it requires two hands; During pretests, participants were not able to reach every key in the 1-handed Phone-Landscape conditions

Hands	Device	Orientation	User Maintained	Once	Swipe
1	Phone	Portrait	2 Hands Required	1-handed part	
		Landscape		Physical Constraints	
	Tablet	Portrait			
		Landscape			
2	Phone	Portrait	2-handed part		
		Landscape			
	Tablet	Portrait			
		Landscape			

4.3.2 One-Handed Part. For the one-handed part, participants were invited to hold, with their preferred hand, a Phone in Portrait orientation only. Tablet device and Landscape orientations were discarded either for physical constrains or because two hands were required for comfortable interaction. Similarly, *UM* was excluded as it requires two simultaneous contact points which cannot be achieved in these configurations. This results in a within-subject design with *Input Method* as a two-level { Once, Swipe } independent variable, fully counterbalanced across participants. In total, a participant would perform 2 Input Methods ×[(1 training block × 6 stimulus) + (3 test blocks × 6 stimulus × 2 repetitions)] = 84 trials.

4.3.3 Two-Handed Part. For the two-handed part, we have a 3 × 2 × 2 within-subject design with three independent variables: *Input method { UM, Once, Swipe }*, *Device { Phone, Tablet }* and *Orientation { Landscape, Portrait }*. Participants were instructed to hold the device with their non-preferred hand and were instructed they could use both hands to perform selection. We did not restrict the specific hand posture, as long as the device was held at least with the non-preferred hand. Input method was counterbalanced using Latin Square, and Device and Orientation were collapsed into four conditions counterbalanced using Latin Square (Phone-Landscape, Phone-Portrait, Tablet-Landscape, Tablet-Portrait). A participant would perform 3 input methods × 2 devices × 2 orientations ×[(1 training block × 6 stimulus) + (3 test blocks × 6 stimulus × 2 repetitions)] = 504 trials in the two-handed part.

4.3.4 Dependent Variables. We measured accuracy and command selection time. A trial was considered successful if the participant was able to select the correct target on first attempt. Selection time was measured as the time between the stimulus presentation until the user finished the selection (release of hotkey for UM and Once, end of the slide gesture for Swipe). The modifier key recorded was the last one used before finishing the trial (left or right). For each individual combination of Technique × Device × Orientation, we also measured subjective feedback on a 1-5 Likert Scale (1: strongly disagree, 5: strongly agree) on their overall opinion for the technique ("I liked using this technique"), ease of use, comfort, perceived speed and perceived accuracy.

Each participant spent approximately 45 minutes to complete the experiment and were allowed to take breaks between each block. In total, we recorded 12 participants × 84 trials in one-handed + 504 trials in two-handed = 7056 trials overall.

4.4 Results

We conducted two separate statistical analyses for one-handed and two-handed parts. After gathering data, we removed outliers (data points above average + 3 standard deviations). We then checked the distribution of our data and found it normally distributed. For the one-handed part, we used paired t-test for accuracy and time (as we only had two Input Methods), and Wilcoxon signed-rank test for subjective measurement. For the two-handed part, we used three-way ANOVA with repeated measures for accuracy and time, with pairwise t-tests with Bonferroni corrections for post-hoc comparisons. We applied Greenhouse-Geisser sphericity correction when needed, which corrects both p-values and the reported degrees of freedom. For subjective measurement, we used Friedman test, with pairwise Wilcoxon signed-rank tests with Bonferroni corrections for post-hoc comparisons. We do not compare performance between one-handed and two-handed parts, as the design and number of trials differ. The 95% confidence intervals shown on the Figures are computed using the z-scores with the following formula: $1.96 \times \frac{SD}{\sqrt{n}}$.

4.4.1 One-handed Part. Participants performed the task rather quickly ($M = 1.09s$) and accurately ($M = 96.2\%$) as seen in Figure 5. While average command selection time was lower with *Once* ($M = 1.04s$) compared to *Swipe* ($M = 1.13s$), the difference was not found significant ($p = .10$). However, we found a significant effect of *Input Method* on accuracy ($t(11) = 5.82, p < .001$), as our participants reached higher accuracy with *Once* ($M = 99.1\%$) than with *Swipe* ($M = 93.3\%$). In terms of subjective preferences, we did not observe any effect of *Input Methods* on any question, with an overall neutral to slightly positive assessment for both methods ($\frac{3.1}{5}$ to $\frac{3.75}{5}$).

4.4.2 Two-handed Part. Time. We found a significant main effect of *Input Method* on time ($F_{2,22} = 25.2, p < .0001, \eta^2 = 0.14$). Participants were faster with *Once* ($M = 0.85s$) than with Swipe

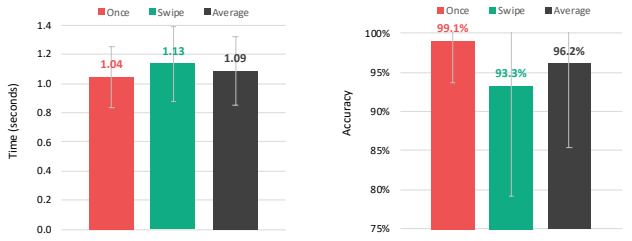

Figure 5: Selection time (Left) and accuracy (Right) performance for the one-handed conditions. Error bars show 95% confidence intervals. Accuracy ordinate axis starts at 75%.

($M = 1.04s, p < .0001$) and *UM* ($M = 0.98s, p < .0001$). We also found a significant main effect of *Orientation* ($F_{1,11} = 10.44, p < .001, \eta^2 = 0.02$), with a slightly lower average command selection time in *Portrait* conditions ($M = 0.92s$ vs. $M = 0.98s$ for *Landscape*). Performance was very similar on Phone and Tablet, with no effect of *Device* ($p = .18$). We also found a significant *Device* × *Orientation* interaction effect on time ($F_{1,11} = 23.58, p < .01, \eta^2 = 0.03$), with *Portrait* being overall faster on *Phone* and *Landscape* on *Tablet*. There was a significant *Device* × *Input Method* interaction effect on time ($F_{2,22} = 7.43, p < .01, \eta^2 = 0.04$), with *Swipe* performing better on *Tablet* than *Phone*, and *UM* following an opposite pattern (Figure 6). We did not find any other interaction effect (all $p > .05$).

Accuracy. Input Method had a significant effect on accuracy ($F_{1.15,12.69} = 19.16, p < .0001, \eta^2 = 0.25$): our participants were significantly less accurate when using *Swipe* ($M = 95.0\%$) compared to *Once* ($M = 99.5\%, p < .0001$) and *UM* ($M = 99.5\%, p < .0001$). The average accuracy was significantly higher ($F_{1,11} = 8.28, p = .015, \eta^2 = 0.05$) on *Tablet* ($M = 98.8\%$) than on the *Phone* ($M = 97.1\%$). We did not find any significant main effect of *Orientation* ($p = .85$) on accuracy. We also observed a significant *Device* × *Input Method* interaction ($F_{2,22} = 12.36, p < .001, \eta^2 = 0.08$) which is due to the increase of performance for *Swipe* between the *Phone* ($M = 92.6\%$) and *Tablet* configurations ($M = 97.3\%$).

Modifier Keys. Our participants used the Left modifier key more often. For *Once*, they used it a total of 1103 times (vs. 625 for Right). Usage was similar for *Swipe* (1108 Left vs. 620 Right). The trend is even clearer for *User Maintained* (1265 trials vs. 463 Right). This is likely due to the constrained posture on the left hand, where they could easily access the Left key with their thumbs.

Subjective Preferences. After each individual condition of our experiment, we gathered subjective preferences of our users. We originally analysed the data for each *Device* × *Orientation* combination, and found out that the results are very similar for each individual condition, and thus decided to aggregate the results (Table 2). Overall, *Once* received positive scores (scores $> \frac{4}{5}$), while *Swipe* and *UM* received neutral to slightly positive scores.

The results of this experiment are discussed in the final discussion of the paper, along with the results of the next experiment.

Table 2: Score (/5) and analysis of subjective preferences. α shows significant ($p < .05$) pairwise differences.

Question	$\chi^2(2)$	p-value	Score (Once)	Score (UM)	Score (Swipe)
Like	9.52	$< .01$	4.13^{α}	3.33^{α}	3.17
Ease of Use	8.77	$< .05$	4.17^{α}	3.54^{α}	3.46
Comfort	6.43	$< .05$	4.13^{α}	3.46^{α}	3.46
Speed	9.91	$< .01$	4.17^{α}	3.38^{α}	3.25
Accuracy	8.33	$< .01$	4.42^{α}	4.33	3.25^{α}

5 EXPERIMENT 2: USAGE OF INPUT METHODS

The previous study investigated the overall performance of each SoftCuts' input methods under various interaction conditions. In this study, our goal is to observe which input method is to be adopted when the user is not constrained and across different activities like sitting, standing and walking. Since the 3 input methods are compatible with one another, the users have the freedom to use the one they want depending on their preferences for each specific scenario. We are interested to see how the results from Experiment 1 would translate when evaluated in a more realistic environment.

5.1 Participants and Apparatus

Twelve participants (all right-handed, 5 female), aged 21 to 28 ($M = 24.2$, $SD = 2.72$) were recruited from the university community. They received the equivalent of 7.5 USD for their participation. None of these participants took part in the previous experiment. Only one participant had never used a tablet prior, and we used the same phone and tablet as that in Experiment 1. In addition, some conditions required the use of a treadmill (see Figure 8) to simulate a consistent walking speed of 2.5 km/h as in previous works [23, 42].

5.2 Procedure

Participants began with completing a questionnaire about their demographic information. They were then briefed on the experiment and started the practice phase. During this practice phase, participants were asked to perform 16 command selections with each *Input Method* while seated down. The command selection was done with the *Phone* and *Tablet*, each *2-handed* and in *Portrait* mode. The order of presentation of the three *Input Methods* was fully counterbalanced across participants to avoid any bias towards one specific *Input Method*. The goal of the practice phase was to give participants a chance to familiarize with each input method before starting the experiment.

After practising, participants were briefed on the actual experiment, in which they had to complete three blocks per individual condition (see Design section). Each block has 16 trials (twice each target, in random order). Participants were instructed to perform the command selection with the *Input Method* they preferred and would rather use in real-life cases. They were also made aware that they were free to change their choice of *Input Method* during the experiment as they wished. After finishing all conditions, participants were asked to explain their choices of *Input Method* to the experimenter in a short semi-structured interview.

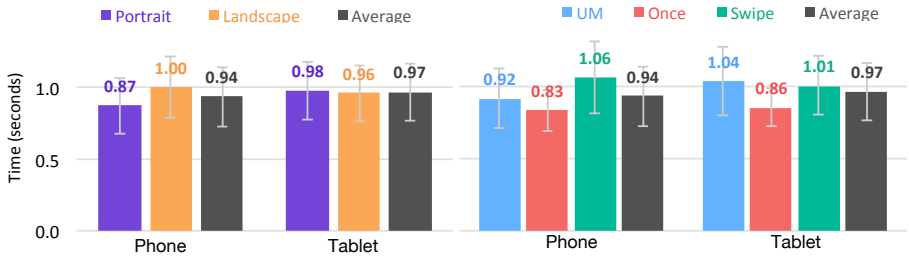

Figure 6: Selection times (Left) for each orientation and (Right) each input method, on both phone and tablet while using 2 hands. Error bars show 95% confidence intervals.

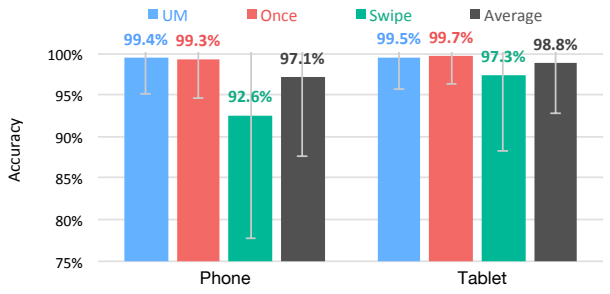

Figure 7: Accuracy performance for individual input method on both phone and tablet for the two-handed conditions. Error bars show 95% confidence intervals. Ordinate axis starts at 75%

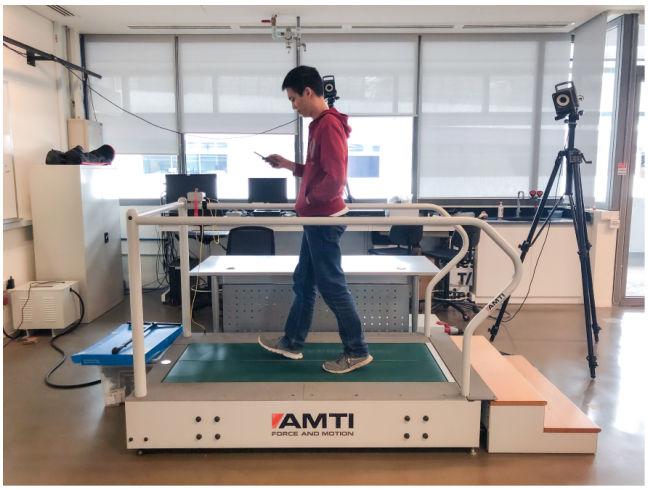

Figure 8: Experimental setup for the treadmill conditions

5.3 Soft Keyboard Layout

Similar to Experiment 1, the soft keyboard used in this study has command keys located at its bottom-left and right corners (Figure 9). When either of the command keys was hit, the soft keyboard will be displayed instantly. Keys with associated commands would be rendered with both its icon and name while keys without associated commands would be greyed out. This graphical representation is different from both commercial solutions that either only highlight the key [43] or display the name at the top of the key (Microsoft Surface) because we leverage on the ability of icons to deliver efficient guiding cues for visual search [7] and conveying command meaning [17]. We chose countries from different continents of the world that have consistent flag shapes, similar to previous studies [18, 33]. The layout for the 16 commands is based on the 16 most common hotkeys on MS Word, which we then mirrored on vertical axis to preserve spatial distribution and have a realistic distribution of commands based on an existing layout.

5.4 Task and Stimulus

Each trial begins with the name of a target command displayed on top of the screen. Then, participants would press the modifier key and select the corresponding command key using any of the three available *Input Methods*. Upon selection, either a positive (in green) or negative (in red) feedback would highlight the selected key for 500 milliseconds, before proceeding to the next trial. For each participant, 8 of the 16 commands were randomly used as stimuli, similar to FastTap [20]. To ensure similar spatial distribution of selected commands, 2 out of the 8 were from the top row and 3 from each of the middle and bottom row. We also avoided mnemonic associations between keys and countries, e.g. "A" for Australia.

5.5 Design

Our initial design space for this experiment includes: Activity {Sitting, Standing, Walking}, Device {Phone, Tablet}, Handedness {1-handed, 2-handed}, Orientation {Landscape, Portrait}. Complete factorial exploration of this design space would lead to a total of 3 $\times$ 2 $\times$ 2 $\times$ 2 = 24 combinations. Participants were asked to hold both devices similar to Experiment 1. However, to keep the experiment within a manageable time frame, we decided to prioritize some factors depending on the device used, and thus discarded some factors for either *Phone* or *Tablet* conditions.

For the *Phone* conditions, we deemed *Handedness* as more important than *Orientation*, as one- and two-handed conditions may yield different results. On the other hand, we did not envision a lot of scenarios with physical activity where landscape orientation would be extensively used and thus decided to set the *Orientation* to

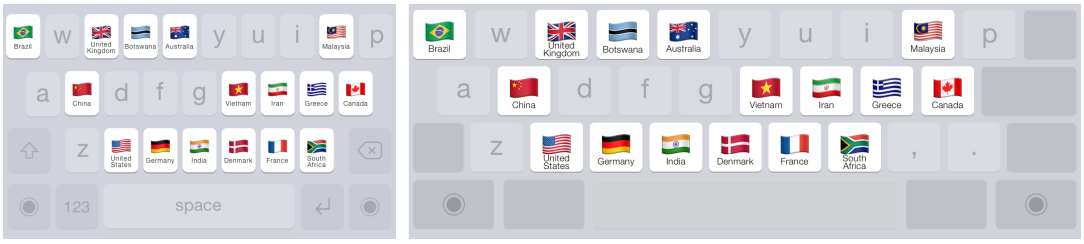

Figure 9: Layout for our second experiment, each shortcut is a country name with its flag. Left: layout on a phone in portrait mode, Right: layout on the tablet in landscape mode.

Portrait mode and included *Handedness* in our design. For the *Tablet* conditions, we assumed that most interactions with a tablet would be two-handed while in motion or simply because supporting the weight with only 1 hand can result in fatigue. As such, we set the *Handedness* to *2-handed* and included *Orientation* in our design. As such, the *Phone* and *Tablet* conditions are not directly comparable as the impact of *Handedness* and *Orientation* is different. We will thus conduct separate analysis for each device.

In this experiment, we measure the usage of each *Input Method* for our different conditions. For each possible *Input Method*, we count the number of trials where command selection was performed using that specific method. Simultaneously, we also measured the corresponding time taken and accuracy (i.e. whether the right command was selected) for each trial. The order of presentation of *Device*, *Handedness* (Phone) and *Orientation* (Tablet) was fully counterbalanced across participants, while *Activity* was counterbalanced using Latin Square.

Each participant took approximately one hour to complete the experiment. Participants were allowed to take breaks between conditions. In summary, we recorded 12 participants × 3 activities × 2 devices × 2 handedness OR orientation × 3 blocks × 8 commands × 2 repetitions = 6912 trials in total.

5.6 Statistical Analysis

The goal of this experiment is to find out which input method participants prefer to use, thus we simply report the usage frequency from our experiment. The results are overall clear, with one technique being chosen most of the time. We were also interested in finding out whether Activity, Handedness and Orientation have an impact on the usage frequency. To do so, we performed a multinomial logistic regression with R by giving numerical values to our activity levels (1: sitting, 2: standing, 3: walking) and Orientation (1: portrait, 2: landscape). We separated the analysis into three parts: one for Phone 1-Handed (as UM is not usable here), one for Phone 2-Handed, and one for Tablet.

From our results, We then compared the time and accuracy performance. However, given the small number of samples for *Swipe* and *User Maintained*, we only performed statistical analysis for time and accuracy for the *Once* method, using ANOVA.

5.7 Results

5.7.1 Usage Frequency. Overall, our users mainly used *Once*, with this method used for 86.2% of selections. Its usage proportion was at least 76.6% (Phone 2-Handed Sitting) and at most 91.3% (Tablet Landscape Walking). The usage frequency of each technique across the different conditions is shown in Figure 10.

Phone 1-Handed. Participants performed most of the trials using *Once* (84.3% of selections), and used *Swipe* for the remaining trials (15.7%). Our analysis shows that *Activity* had an impact on the usage frequency of both techniques (both $p < .001$): the usage of *Swipe* and *Once* varies with *Activity*, from 18.2%-82.8% (resp. Swipe and Once) sitting to 16.3%-83.7% standing and 12.5%-87.5% walking.

Phone 2-Handed. Once was again the most frequently used (82.5% of selections), with a significant effect of *Activity* ($p < .001$) with usage ranging from 76.6% while sitting, to 83.5% while walking and 87.2% while standing. Swipe was the second most used at a steady 11.7% of usage, ranging from 11.1% (sitting) to 12.2% (walking) with no effect of *Activity* ($p > .05$). Finally, *UM* was used for 5.9% of selections, and was strongly impacted by *Activity* ($p < .001$) with little to no usage while standing (1%), rather limited while walking (5.9% usage) and to a level comparable to *Swipe* while sitting (11.3%).

Tablet. Once was overall used for 89% of the selections, and its usage was not impacted by *Orientation* or *Activity* (both $p > .05$). *UM* was the second most chosen input method, used for 10% of the selections consistently across conditions with no visible effect of *Orientation* or *Activity* (both $p > .05$). Swipe was only used for 0.98% of the trials with no visible effect of *Activity* on its usage. However, we did notice an effect of *Orientation* on *Swipe* usage ($p < .01$): our participants nearly did not use Swipe in Portrait mode (0.3% of selections) and used it a bit more in Landscape (1.6% of selections).

Individual Participants Usage. Among our participants, we did notice that one specific participant (P8) used *Swipe* for 98.9% of their trials on the *Phone* and *UM* for 99.3% of their trials on *Tablet*. The participant reported that they did not like the *Once* method overall and that they were used to *Swipe* for typing on soft keyboards on their phone. This suggests that exposure to input methods may have an impact on usage frequency. Our other participants showed a consistent pattern of relying mostly on *Once* for 90% selections or more and usually trying out the other techniques across conditions.

5.7.2 Time and Accuracy. We also analysed the objective performance by filtering trials that were completed using *Once*. We could

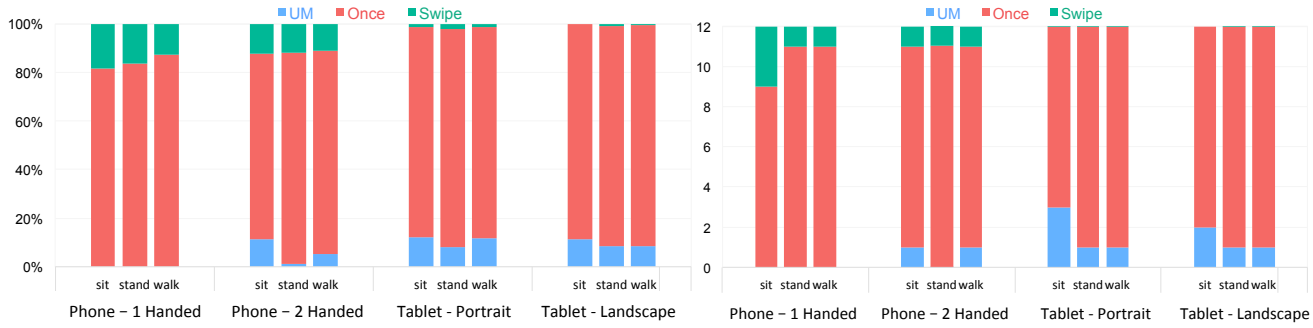

Figure 10: Distribution of each *Input Method* (UM, Once, Swipe) across 4 conditions between device, handedness, orientation while sitting, standing and walking. (Left) is derived from actual usage during the experiment while (Right) is the declared preference at the end of each condition.

not perform the analysis for the other two techniques because of the lack of data available for some conditions, as these techniques were seldom used (e.g. *Swipe* on *Tablet* or *UM* on *Phone*). We performed one analysis for each device because as mentioned in the Design section above, we included *Handedness* for phone's design space and *Orientation* for tablet's.

Phone. In terms of time, we found significant effect of *Handedness* ($F_{1,10} = 9.03, p < .05, \eta^2 = 0.06$) on phone's performance. It took participants on average 1.68s to complete a trial on a phone using 1 hand, and 1.513s using 2 hands. We did not find any effect of *Activity* or interaction (both $p > .05$). For accuracy, we did not observe any effect of *Handedness*, *Activity* or interactions, with an average accuracy of 98.4%.

Tablet. For the tablet conditions, there was no significant effect of *Activity*, *Orientation* and no interaction on time (all $p > .05$). A trial would on average last for 1.509s to complete, which is very similar to that of phone's 2-handed performance. In terms of accuracy, there was no significant effect ($p > .05$) of *Activity*, *Orientation* and no interaction, with participants reaching 98.6% accuracy on average.

The time and accuracy results seem to be in line with that of Experiment 1, where the performance of *Once* is comparable between on Phone-2 handed (0.85s) and on Tablet (0.89s). The task in experiment 1 was a simple pointing task, while the task in Experiment 2 required the user to perform a visual search to identify the correct command. The effect of *Handedness* on time for the phone condition seems consistent with the average performance observed in Experiment 1, where users were overall faster in 2-handed mode (0.85s) vs. 1-handed mode (1.04s). An interesting result from Experiment 2 is that the performance of *Once* does not seem to be affected by *Activity*, making *Once* a robust input method for real life scenarios.

6 DISCUSSION

We now discuss our results, with the strengths of each input method and what these results mean for SoftCuts. For each input method, we also discuss how the technique could be adapted to perform multiple hotkey selections sequentially.

6.1 Is Once the best input method for SoftCuts?

In our first study, *Once* achieved the overall highest performance in terms of time, accuracy and preference. *Once* significantly stands out, yielding the lowest command selection time both on tablet and phone, regardless of device orientation and they were held. Participants also achieved near-perfect accuracy of 98-99% in both experiments. From experiment 2, we found out that *Once* was chosen as an input method near 4 out of 5 trials, making it the most used input method, no matter the device, handedness or tablet orientation. More importantly, the performance of this method did not seem to be affected by *Activity*, making *Once* the best input method.

Several participants (P1,5,6,7) shared the similar sentiment that *Once* is "versatile" because of its two discrete steps, unlike in *UM* having to communicate between 2 finger actions, which requires more "cognitive effort to use" (P12). That being said, we must stress that both of our experiments required users to perform a single command selection per trial, which appears to be the best case scenario for this technique. Indeed, *Once* relies on a pseudo-mode (unlike *UM* which relies on a user-maintained mode). Therefore, it may be less efficient and more tedious to use for performing several command selections at once. This is given that the keyboard would switch back to text entry mode after each command selection, hence requiring users to reactivate command mode for each subsequent command selection. One way around that would be to allow users to double-tap on the modifier key which would lock the mode to command selection, the same way that one can double-tap the Shift key on a soft keyboard to maintain the upper case mode.

As a very first investigation of SoftCuts, we had to make experimental decisions and trade-offs to keep the length of our studies under acceptable duration. Future work should investigate the impact of the number of commands to be selected per trial on input method performance and usage.

6.2 Swipe as a suitable Phone input method

Swipe was a compelling case to investigate. In experiment 1, we showed that its performance was objectively worse than other input methods. The gap was more significant on the phone, where we observed a lower accuracy overall (around 92% on average in both 1- and 2-handed), as well as a larger selection time. The performance of *Swipe* on the tablet was closer to the other techniques, and thus, we expected to see better adoption of *Swipe* on tablets. However, it appears that users simply did not want to use *Swipe* on the tablet, as it was used for around 1% of the selections, and even less in landscape mode. P4,5,7,8,11,12 found dragging across the larger screen area of the tablet was "challenging" and "uncomfortable". Participants reported that swiping was too "tedious on the tablet", usually because of "the long distance that [one] needs to swipe" (P4). P8 also reported that swiping allowed them to "change their decision before releasing the gesture at the right key", suggesting that *Swipe* allows user to partially recover from erroneous selections, as long as they did not release their finger.

Despite these issues, Swipe was the second most used input methods for phones, used for 15.7% of the selections in Phone 1-handed, and 11.7% of the selections in Phone 2-handed. One participant even exclusively used *Swipe* citing their large usage of swipe based text entry methods and their strong dislike of *Once*. All users did perform part of their selections using *Swipe* on the phone nonetheless. This suggests that *Swipe* could still be the primary input method on the phone for certain categories of users, especially in 1-handed scenarios, despite its objectively worse performance. It is also worth noting that participants tended to rely on *Swipe* more on sitting and standing positions, while the selection was deemed harder while walking. *Swipe*, by default, does not support multiple selections in a row. It could support that using, for example, a dwell mechanism: users willing to select a specific command would stop on it for a short duration, then proceed to the next command. However, this would likely put the technique at a substantial disadvantage compared to *Once* and even more to *UM*.

6.3 User Maintained as a Tablet input method

UM is the only of our input methods that require two hands to operate. However, its performance in Experiment 1 is close to *Once* in terms of speed and accuracy, with a small difference of 100 ms for Phone conditions. On the other hand, its subjective preference was overall the lowest, which led us to believe that *UM* may not be used at all.

We were once again surprised to find out that *UM* was used in Experiment 2. In Tablet mode, specifically, it was chosen around 10% of the time. P5 commented about an ergonomic advantage that they felt with *UM* while supporting the relatively heavy weight of the tablet. They said, "as the thumb holds onto the modifier key at the bottom, a grip action is resembled, hence making it easier and more steady to use the other hand's finger to trigger the (target) key." Another user also performed most of their selection with this method, because of their dislike of *Once*.

Another strong advantage of *UM* over the other techniques is that it readily supports multiple selections, as illustrated in Figure 3. It is important to note that both our experiments featured only a single command selection per trial, putting *UM* in a worst-case scenario. We could thus expect *UM* to be more widely used for productivity tasks, where users select multiple commands at once.

6.4 SoftCuts in real world

Both experiments show that users are able to quickly select commands, with an average time around 0.95s in Experiment 1 (pointing only) vs. 1.5s for *Once* in Experiment 2. Accuracy was also extremely high and users enjoyed using the system as seen in the high score of the subjective preferences. This makes SoftCuts a good candidate for command selection technique on both tablet and phones. Our vision for SoftCuts is illustrated in Figure 1.

Input Methods. A minimal way to implement SoftCuts on mobile devices is to provide *Once* as an input method. However, as highlighted above, some users are likely willing to use *Swipe* on their phone, and *UM* already supports multiple command selection. Based on our state machine presented in Figure 3, the three input methods are compatible with each other, allowing designers to simply implement all three of them without any interference.

Advantages over physical hotkeys. In addition, SoftCuts make good use of the space used on a soft keyboard. Compared to physical hotkeys, where users need to memorize the mapping between hotkeys and commands, SoftCuts display the name of the command with an icon, making it easier to do command selection even with low familiarity with the mapping of the current application.

Advantages over other command selection techniques. Compared to other command selection techniques, SoftCuts also potentially leverage users' prior knowledge of hotkeys on desktop computers and can be implemented on any applications, making it a generic and consistent command selection technique across devices. Implementing SoftCuts comes at the small cost of adding a modifier key (either Control or Command) at the bottom left part of the soft keyboard. However, future work is needed to further investigate on the discoverability of SoftCuts.

Keyboard-less scenarios. While adding one or two modifier key(s) on a soft keyboard layout is rather straightforward for any scenarios where the keyboard is displayed, the same modifier key(s) could be displayed semi-transparently in keyboard-less scenarios (Figure 1). This would partly occlude a small part of the screen, but make the technique extremely salient to the user, as it would lead them to press the key to try SoftCuts. Throughout both experiments, multiple participants suggesting the use of either semi-transparent or fully-transparent modifiers keys for keyboard-less scenarios. The idea of a fully-transparent key [49] was for expert users, as they would remember the overall location of the modifier key on the keyboard layout and would be able to start the selection without the need to look for the exact position of the modifier key. In future work, we would like to evaluate the performance of SoftCuts while carrying a real-world task including keyboard-less applications.

7 CONCLUSION

We explored the concept of soft keyboard shortcuts/hotkeys (SoftCuts) as a generic command selection mechanism for touch-based

devices. We showed that its three input methods yielded high performance, with Once being the best for any device, orientation, handedness, and also most preferred by participants. Interestingly, we found that some participants preferred *Swipe* on phone despite its objectively worse performance on every aspect. As future work, we would like to investigate how to optimize SoftCuts' layout as well as potential skill transfer either from physical keyboards to soft keyboards, or vice versa.

ACKNOWLEDGMENTS

This work was partially supported by the Agence Nationale de la Recherche (Discovery, ANR-19-CE33-0006) and by the Singapore Ministry of Education and Singapore University of Technology and Design (SUTD) Start-up Research Grant (T1SRIS18141).

REFERENCES

[1] Jessalyn Alvina, Andrea Bunt, Parmit K Chilana, Sylvain Malacria, and Joanna McGrenere. 2020. Where is that Feature? Designing for Cross-Device Software Learnability. In *Proceedings of the 2020 conference on Designing Interactive Systems (DIS '20)*. Association for Computing Machinery, New York, NY, USA. https://doi.org/10.1145/3357236.3395506

[2] Jessalyn Alvina, Carla F. Griggio, Xiaojun Bi, and Wendy E. Mackay. 2017. CommandBoard: Creating a General-Purpose Command Gesture Input Space for Soft Keyboard *(UIST '17)*. ACM, New York, NY, USA, 17–28. https://doi.org/10.1145/3126594.3126639

[3] Toshiyuki Ando, Toshiya Isomoto, Buntarou Shizuki, and Shin Takahashi. 2019. One-handed Rapid Text Selection and Command Execution Method for Smartphones *(CHI EA '19)*. ACM, New York, NY, USA, Article LBW0224, 6 pages. https://doi.org/10.1145/3290607.3312850

[4] Caroline Appert and Shumin Zhai. 2009. Using Strokes As Command Shortcuts: Cognitive Benefits and Toolkit Support *(CHI '09)*. ACM, New York, NY, USA, 2289–2298. https://doi.org/10.1145/1518701.1519052

[5] Oscar Kin-Chung Au, Kira Yip-Wo Yeung, and Jacky Kit Cheung. 2016. DownChord and UpChord: A New Style of Keyboard Shortcuts Based on Simultaneous Key-down and Key-up Events *(ChineseCHI2016)*. ACM, New York, NY, USA, Article 3, 10 pages. https://doi.org/10.1145/2948708.2948715

[6] Jeff Avery and Edward Lank. 2016. Surveying Expert-Level Gesture Use and Adoption on Multi-Touch Tablets *(DIS '16)*. ACM, New York, NY, USA, 577–581. https://doi.org/10.1145/2901790.2901895

[7] Gilles Bailly, Eric Lecolinet, and Laurence Nigay. 2016. Visual Menu Techniques. *ACM Comput. Surv.* 49, 4, Article 60 (Dec. 2016), 41 pages. https://doi.org/10.1145/3002171

[8] Gilles Bailly, Thomas Pietrzak, Jonathan Deber, and Daniel J. Wigdor. 2013. Métamorphe: Augmenting Hotkey Usage with Actuated Keys *(CHI '13)*. ACM, New York, NY, USA, 563–572. https://doi.org/10.1145/2470654.2470734

[9] Olivier Bau and Wendy E. Mackay. 2008. OctoPocus: A Dynamic Guide for Learning Gesture-based Command Sets *(UIST '08)*. ACM, New York, NY, USA, 37–46. https://doi.org/10.1145/1449715.1449724

[10] Suresh K. Bhavnani and Bonnie E. John. 2000. The Strategic Use of Complex Computer Systems. *Hum.-Comput. Interact.* 15, 2 (Sept. 2000), 107–137. https://doi.org/10.1207/S15327051HCI1523_3

[11] Stuart K. Card, Thomas P. Moran, and Allen Newell. 1980. The Keystroke-level Model for User Performance Time with Interactive Systems. *Commun. ACM* 23, 7 (July 1980), 396–410. https://doi.org/10.1145/358886.358895

[12] Karim El Batran and Mark D. Dunlop. 2014. Enhancing KLM (Keystroke-Level Model) to Fit Touch Screen Mobile Devices *(MobileHCI '14)*. Association for Computing Machinery, New York, NY, USA, 283–286. https://doi.org/10.1145/2628363.2628385

[13] Katherine Fennedy and Hyowon Lee. 2019. Multitouch Keyboard Revisited: Enhancing Moded Interaction Through Redesigning Structure and Switching Techniques *(CHIuXiD'19)*. Association for Computing Machinery, New York, NY, USA, 36–45. https://doi.org/10.1145/3328243.3328249

[14] Jérémie Francone, Gilles Bailly, Eric Lecolinet, Nadine Mandran, and Laurence Nigay. 2010. Wavelet Menus on Handheld Devices: Stacking Metaphor for Novice Mode and Eyes-free Selection for Expert Mode *(AVI '10)*. ACM, New York, NY, USA, 173–180. https://doi.org/10.1145/1842993.1843025

[15] Jeremie Francone, Gilles Bailly, Laurence Nigay, and Eric Lecolinet. 2009. Wavelet Menus: A Stacking Metaphor for Adapting Marking Menus to Mobile Devices *(MobileHCI '09)*. ACM, New York, NY, USA, Article 49, 4 pages. https://doi.org/10.1145/1613858.1613919

[16] Emilien Ghomi, Stéphane Huot, Olivier Bau, Michel Beaudouin-Lafon, and Wendy E. Mackay. 2013. ArpèGe: Learning Multitouch Chord Gestures Vocabularies *(ITS '13)*. ACM, New York, NY, USA, 209–218. https://doi.org/10.1145/2512349.2512795

[17] Emmanouil Giannisakis, Gilles Bailly, Sylvain Malacria, and Fanny Chevalier. 2017. IconHK: Using Toolbar Button Icons to Communicate Keyboard Shortcuts *(CHI '17)*. ACM, New York, NY, USA, 4715–4726. https://doi.org/10.1145/3025453.3025595

[18] Alix Goguey, Sylvain Malacria, Andy Cockburn, and Carl Gutwin. 2019. Reducing Error Aversion to Support Novice-to-Expert Transitions with FastTap *(IHM '19)*. Association for Computing Machinery, New York, NY, USA, Article Article 1, 10 pages. https://doi.org/10.1145/3366550.3372247

[19] Tovi Grossman, Pierre Dragicevic, and Ravin Balakrishnan. 2007. Strategies for Accelerating On-line Learning of Hotkeys *(CHI '07)*. ACM, New York, NY, USA, 1591–1600. https://doi.org/10.1145/1240624.1240865

[20] Carl Gutwin, Andy Cockburn, Joey Scarr, Sylvain Malacria, and Scott C. Olson. 2014. Faster Command Selection on Tablets with FastTap *(CHI '14)*. ACM, New York, NY, USA, 2617–2626. https://doi.org/10.1145/2556288.2557136

[21] Jay Henderson, Sylvain Malacria, Mathieu Nancel, and Edward Lank. 2020. Investigating the Necessity of Delay in Marking Menu Invocation *(CHI '20)*. Association for Computing Machinery, New York, NY, USA, 1–13. https://doi.org/10.1145/3313831.3376296

[22] Seongkook Heo, Jingun Jung, and Geehyuk Lee. 2016. MelodicTap: Fingering Hotkey for Touch Tablets *(OzCHI '16)*. ACM, New York, NY, USA, 396–400. https://doi.org/10.1145/3010915.3010993

[23] Seungwoo Je, Minkyeong Lee, Yoonji Kim, Liwei Chan, Xing-Dong Yang, and Andrea Bianchi. 2018. PokeRing: Notifications by Poking Around the Finger *(CHI '18)*. Association for Computing Machinery, New York, NY, USA, Article Paper 542, 10 pages. https://doi.org/10.1145/3173574.3174116

[24] Tero Jokela, Jarno Ojala, and Thomas Olsson. 2015. A Diary Study on Combining Multiple Information Devices in Everyday Activities and Tasks *(CHI '15)*. Association for Computing Machinery, New York, NY, USA, 3903–3912. https://doi.org/10.1145/2702123.2702211

[25] Kenrick Kin, Björn Hartmann, and Maneesh Agrawala. 2011. Two-handed Marking Menus for Multitouch Devices. *ACM Trans. Comput.-Hum. Interact.* 18, 3, Article 16 (Aug. 2011), 23 pages. https://doi.org/10.1145/1993060.1993066

[26] Per-Ola Kristensson and Shumin Zhai. 2004. SHARK2: A Large Vocabulary Shorthand Writing System for Pen-based Computers *(UIST '04)*. ACM, New York, NY, USA, 43–52. https://doi.org/10.1145/1029632.1029640

[27] Per Ola Kristensson and Shumin Zhai. 2007. Command Strokes with and Without Preview: Using Pen Gestures on Keyboard for Command Selection *(CHI '07)*. ACM, New York, NY, USA, 1137–1146. https://doi.org/10.1145/1240624.1240797

[28] Gordon Kurtenbach and William Buxton. 1994. User Learning and Performance with Marking Menus *(CHI '94)*. ACM, New York, NY, USA, 258–264. https://doi.org/10.1145/191666.191759

[29] Benjamin Lafreniere, Carl Gutwin, and Andy Cockburn. 2017. Investigating the Post-Training Persistence of Expert Interaction Techniques. *ACM Trans. Comput.-Hum. Interact.* 24, 4, Article 29 (Aug. 2017), 46 pages. https://doi.org/10.1145/3119928

[30] Benjamin Lafreniere, Carl Gutwin, Andy Cockburn, and Tovi Grossman. 2016. Faster Command Selection on Touchscreen Watches *(CHI '16)*. ACM, New York, NY, USA, 4663–4674. https://doi.org/10.1145/2858036.2858166

[31] David Lane, H. Napier, S. Peres, and Aniko Sandor. 2005. Hidden Costs of Graphical User Interfaces: Failure to Make the Transition from Menus and Icon Toolbars to Keyboard Shortcuts. *Int. J. Hum. Comput. Interaction* 18 (05 2005), 133–144. https://doi.org/10.1207/s15327590ijhc1802_1

[32] G. Julian Lepinski, Tovi Grossman, and George Fitzmaurice. 2010. The Design and Evaluation of Multitouch Marking Menus *(CHI '10)*. ACM, New York, NY, USA, 2233–2242. https://doi.org/10.1145/1753326.1753663

[33] Sylvain Malacria, Gilles Bailly, Joel Harrison, Andy Cockburn, and Carl Gutwin. 2013. Promoting Hotkey Use Through Rehearsal with ExposeHK *(CHI '13)*. ACM, New York, NY, USA, 573–582. https://doi.org/10.1145/2470654.2470735

[34] Sylvain Malacria, Joey Scarr, Andy Cockburn, Carl Gutwin, and Tovi Grossman. 2013. Skillometers: Reflective Widgets That Motivate and Help Users to Improve Performance *(UIST '13)*. ACM, New York, NY, USA, 321–330. https://doi.org/10.1145/2501988.2501996

[35] Craig S. Miller, Svetlin Denkov, and Richard C. Omanson. 2011. Categorization Costs for Hierarchical Keyboard Commands *(CHI '11)*. ACM, New York, NY, USA, 2765–2768. https://doi.org/10.1145/1978942.1979351

[36] Donald A. Norman. 2010. Natural User Interfaces Are Not Natural. *Interactions* 17, 3 (May 2010), 6–10. https://doi.org/10.1145/1744161.1744163

[37] Donald A. Norman and Jakob Nielsen. 2010. Gestural Interfaces: A Step Backward in Usability. *Interactions* 17, 5 (Sept. 2010), 46–49. https://doi.org/10.1145/1836216.1836228

[38] Nuance. 2017. What are the most common Swype gestures? https://nuancemobility.zendesk.com/hc/en-us/articles/212253703-What-are-the-most-common-Swype-gestures-.

[39] Daniel L. Odell, Richard C. Davis, Andrew Smith, and Paul K. Wright. 2004. Toolglasses, Marking Menus, and Hotkeys: A Comparison of One and Two-handed Command Selection Techniques *(GI '04)*. Canadian Human-Computer Communications Society, School of Computer Science, University of Waterloo, Waterloo, Ontario, Canada, 17–24. http://dl.acm.org/citation.cfm?id=1006058.1006061

[40] PandaSage1221. [n.d.]. *Typing on Windows 8 / Windows RT Onscreen Touch Keyboard + Features*. Youtube. https://youtu.be/Dep_O1508Ug

[41] Nicole Pong and Sylvain Malacria. 2019. Awareness, Usage and Discovery of Swipe-revealed Hidden Widgets in iOS. In *Proceedings of the 2019 ACM International Conference on Interactive Surfaces and Spaces (ISS '19)*. ACM, New York, NY, USA.

[42] Thijs Roumen, Simon T. Perrault, and Shengdong Zhao. 2015. NotiRing: A Comparative Study of Notification Channels for Wearable Interactive Rings *(CHI '15)*. Association for Computing Machinery, New York, NY, USA, 2497–2500. https://doi.org/10.1145/2702123.2702350

[43] Samsung. [n.d.]. Can I use Ctrl + C, Ctrl + V and other shortcuts on Galaxy Tab S like I would on a PC? https://www.samsung.com/uk/support/mobile-devices/can-i-use-ctrl-c-ctrl-v-and-other-shortcuts-on-galaxy-tab-s-like-i-would-on-a-pc/.

[44] Katherine Schramm, Carl Gutwin, and Andy Cockburn. 2016. Supporting Transitions to Expertise in Hidden Toolbars *(CHI '16)*. ACM, New York, NY, USA, 4687–4698. https://doi.org/10.1145/2858036.2858412

[45] Art. Lebedev Studio. 2019. Optimus Popularis. https://www.artlebedev.com/optimus/popularis/.

[46] Susanne Tak, Piet Westendorp, and Iris Van Rooij. 2013. Satisficing and the use of keyboard shortcuts: being good enough Is enough? *Interacting with computers* 25, 5 (2013), 404–416.

[47] Jingjie Zheng, Xiaojun Bi, Kun Li, Yang Li, and Shumin Zhai. 2018. M3 Gesture Menu: Design and Experimental Analyses of Marking Menus for Touchscreen Mobile Interaction *(CHI '18)*. ACM, New York, NY, USA, Article 249, 14 pages. https://doi.org/10.1145/3173574.3173823

[48] Jingjie Zheng, Blaine Lewis, Jeff Avery, and Daniel Vogel. 2018. FingerArc and FingerChord: Supporting Novice to Expert Transitions with Guided Finger-Aware Shortcuts *(UIST '18)*. ACM, New York, NY, USA, 347–363. https://doi.org/10.1145/3242587.3242589

[49] Suwen Zhu, Tianyao Luo, Xiaojun Bi, and Shumin Zhai. 2018. Typing on an Invisible Keyboard *(CHI '18)*. Association for Computing Machinery, New York, NY, USA, 1–13. https://doi.org/10.1145/3173574.3174013