Towards a Unified and Efficient Command Selection Mechanism for Touch-Based Devices Using Soft Keyboard Hotkeys

Katherine Fennedy, Angad Srivastava, Sylvain Malacria, Simon T Perrault

Towards a Unified and Efficient Command Selection Mechanism for Touch-Based Devices Using Soft Keyboard Hotkeys

KATHERINE FENNEDY, Singapore University of Technology and Design (SUTD) & NUS-HCI Lab, Singapore
ANGAD SRIVASTAVA, Singapore University of Technology and Design (SUTD) & Yale-NUS College, Singapore
SYLVAIN MALACRIA, Univ. Lille, Inria, CNRS, Centrale Lille, UMR 9189 CRIStAL, France
SIMON T. PERRAULT, Singapore University of Technology and Design (SUTD), Singapore

We advocate for the usage of hotkeys on touch-based devices by capitalising on soft keyboards through four studies. First, we evaluated visual designs and recommended icons with command names for novices while letters with command names for experts. Second, we investigated the discoverability by asking crowdworkers to use our prototype, with some tasks only doable upon successfully discovering the technique. Discovery rates were high regardless of conditions that vary the familiarity and saliency of modifier keys. However, familiarity with desktop hotkeys boosted discoverability. Our third study focused on how prior knowledge of hotkeys could be leveraged and resulted in a 5% selection time improvement and identified the role of spatial memory in retention. Finally, we compared our soft keyboard layout with a grid layout similar to FastTap. The latter offered a 12-16% gain on selection speed, but at a high cost in terms of screen estate and low spatial stability.

CCS Concepts: • **Human-centered computing → Interaction techniques**; **Touch screens**.

Additional Key Words and Phrases: interaction technique, command selection, mobile devices, touch-based devices, keyboard shortcuts, hotkeys

ACM Reference Format:
Katherine Fennedy, Angad Srivastava, Sylvain Malacria, and Simon T. Perrault. . Towards a Unified and Efficient Command Selection Mechanism for Touch-Based Devices Using Soft Keyboard Hotkeys. In *Proceedings of* . ACM, New York, NY, USA, Article 1, 39 pages.

1 INTRODUCTION

Hotkeys (or keyboard shortcuts) are efficient command selection mechanism commonly deployed on desktop systems. They facilitate rapid access to specific commands by pressing a modifier key together with another character key [1]. For instance, commands like cut, copy, and paste are associated with the X, C, and V character key, respectively, and have been established universally.

Unlike desktop systems, touch-based devices usually rely on menus and gestures for command selection. On existing smartphones and tablets, commands like finding words require multiple taps, and essential text-editing commands, like undo, are either not supported or only accessible via "physical" gestures like shaking the device [6]. Moreover, command positions within menu hierarchies vary depending on the application, while gestures are mostly hidden and not signified to the user [63], thus making them difficult to discover [64]. Consider a simple scenario in the Notes

[1]While modifier-less hotkeys exist, typically in graphical applications such as Adobe Photoshop or GIMP, they remain much less frequent than hotkeys relying on modifiers. Therefore, we focus on this common type of hotkey in this paper.

Manuscript submitted to ACM

application on iOS. A user would want to copy a paragraph, find a target word, paste the paragraph after the found word, and then undo it because she is not happy with the result. In this scenario, the user must select the copy and paste command from the contextual menu (revealed by sequentially tapping the paragraph to select it); look for the target word by activating the "Find in Note" command from another menu revealed by tapping on button at the top of the screen; and undo the paste action by shaking the device (a physical gesture that the user has to know beforehand). Interestingly, the same commands can be activated using different interaction paradigms depending on the application. For instance, the Pages and Keynotes apps on iOS have dedicated undo buttons displayed at the top of the interface. On the other hand, Safari offers two methods to find a specific text on a Web page: either 1) by tapping the (*share*[2]) button then select the “Find on Page” command, or 2) by typing the target text in the URL text-field then select the “On this page" result located at the bottom of the list of suggested websites. These limitations take a toll on performance because users must relearn gestures or re-familiarise with different paradigms and menu hierarchies as they switch between applications.

In contrast, these four commands (copy, paste, find and undo) can be activated through the menu bar of most desktop applications. Even better, if the user already knows the corresponding hotkey combinations, they can all be consistently and efficiently activated via keyboard shortcuts. Touch-based devices lack a similar unified mechanism for efficient command selections.

Since soft keyboards are available on virtually every touch-based device, we are motivated to explore the efficacy of integrating keyboard shortcut functionalities onto smartphones and tablets. The fact that a soft keyboard can update what each key displays on demand is indeed an excellent feature to support hotkey-like interaction without requiring users to memorise and recall key mappings. This concept of soft keyboard shortcuts/hotkeys (SoftCuts) can already be found on Microsoft Surface [59] and Samsung Galaxy tablets [66]. Both use different selection mechanisms and visual representation of shortcuts. Recent work by Fennedy et al. [21] investigated three different input methods for SoftCuts. They suggested that *Once*, a method relying on two sequential taps was overall a better choice compared to *User-Maintained* (holding on one key while tapping on a second key, like for desktop keyboard shortcuts) or *Swipe* (sliding gestures). While this work allows us to get a clearer understanding of how to improve input performance for SoftCuts, there are many remaining questions on how to realise the potential of SoftCuts as a unified command selection mechanism for touch-based devices. More specifically, we focus on the impact of the keyboard-based layout and its visual representation on the discoverability and learnability of SoftCuts. We explore these aspects through the following research questions:

1. Which visual representation is preferred and suitable for SoftCuts?
2. Do users spontaneously discover the SoftCuts mechanism, and what is the impact of keyboard visibility on this discovery?
3. How does the icon displayed by the modifier key affect discoverability of SoftCuts?
4. How do users learn command mappings and does prior knowledge help them?
5. How does the keyboard-based layout of SoftCuts impact their performance compared to a grid-based layout supporting larger buttons?

To answer these questions, we conducted four studies. In our first study, we evaluated the design rationale and preference between three visual designs for the on-screen keys, showing either the letter only (D1), the letter and command name (D2), or the command name and an icon (D3), through semi-structured interviews and subjective

[2]Note that this button actually displays the “share” icon without any textual hint

ratings. We found that D3 was most positively received, while D2 was a good alternative for experienced users. In our second study conducted with 182 crowdworkers, we investigated the discoverability of SoftCuts in two scenarios with two different designs for the modifier keys. We used one scenario (note-taking) where the keyboard was always displayed on the screen and another scenario (web-browsing) where the keyboard was hidden by default. For the modifier keys, we used either the ⌘ / Ctrl (familiar to Apple/Windows/Linux users) or a completely new and arbitrary design (⌘). In both scenarios, we asked crowdworkers to perform a series of tasks, some of which could be achieved only by using SoftCuts. Our results revealed a relatively high discovery rate of SoftCuts and that neither familiarity nor saliency of modifier key affects discovery rate. However, being familiar with hotkeys in general boost SoftCuts' discoverability. In our third study, we compared the selection performance and retention between a "realistic" mapping (using potentially known commands) and an abstract mapping (using unknown arbitrary commands). Our results suggest that users rapidly reached an efficient and stable performance level with both mappings, with a small 5% performance gap. Participants also leveraged spatial memory in both conditions, remembering commands' locations even after a week. In our final study, we compared selection performance between the keyboard layout approach for SoftCuts and a grid layout approach displaying larger buttons but occupying the entire screen (similar to FastTap [31] in terms of layout). We learned that while the grid layout was 12-16% faster than the keyboard one, no significant difference was found in accuracy. On the other hand, the keyboard layout supports greater spatial stability than the grid layout while using a much lower screen real estate. Altogether, our results suggest a *discoverable* design for SoftCuts that yields efficient performance and lets users already familiar with hotkeys on other devices capitalise on this knowledge. As such, SoftCuts could be used as a unified command selection mechanism on touch-based devices. It also presents the advantage of homogenising command selection interaction techniques across devices due to its similarity with familiar desktop keyboard shortcuts.

2 RELATED WORK

We first review the key strategies adopted by today's touch-based devices before discussing the potential challenges and benefits of using an alternative command selection mechanism like hotkeys which are common for desktop computers. Merging these two ideas, we then focus on introducing the concept of SoftCuts, its latest commercial and academic developments, and most importantly, the significant gaps that left its full potential unexploited. We also include prior works that support how SoftCuts could maximise both discoverability and cross-device learning of commands.

2.1 Command Selection on Smartphones/Tablets

Touch-based devices rely on two primary modalities to select commands: 1) tap-based and 2) gesture-based. Despite the multitouch capabilities, single-point tap-based interaction is still commonly used today to trigger commands displayed in toolbars or hidden within hamburger menus, a mobile design pattern adopted due to the limited screen space. Such slow interaction is degrading the user experience over time [61]. On the other hand, gestures like swiping [70] and shaking [6] are known for their simple and fast execution, but they are challenging to discover, especially for novices [8, 64]. The seminal Marking Menus [39] adopt a design that supports users progression over time and has inspired the usage of stroke-based gestures as command shortcuts on touch-based devices [5, 23, 37, 44, 78]. However, the mapping between a gestural input and a command output is invisible, inconsistent, and not natural [55], hence prompting users to continuously rehearse and relearn new mappings as they switch between applications or devices. An attempt to alleviate this problem can be found in FastTap [31], a rehearsal-based technique that displays commands in a spatially stable grid layout and triggers them through chorded selections. Spatially consistent interfaces have been shown useful

for command selection [67] and robust to view transformations, especially scaling [68] which makes command selection mechanisms like FastTap interesting for touch-based devices varying in size and format. However, as a rehearsal-based technique, FastTap uses a delay before displaying the menu. Delay not only can increase error rate [32, 46] but also its necessity has been questioned in the context of Marking Menus [32]. On the other hand, SoftCuts is a delay-less technique that relies on a keyboard layout, a spatially consistent interface. To our knowledge, there have not been any studies that assess the performance of command selection between a grid and keyboard layout.

2.2 Optimising Hotkeys Learnability

On desktop computers, hotkeys offer an alternative to the pointer-based selection of commands from toolbars and menubars. However, their established performance benefits [14, 29, 47, 51, 58] come at the cost of having to learn the corresponding key combination beforehand (through paired-associate learning [7]) in order to be able to select the command using a hotkey. As a result, hotkeys are often considered underused [12, 42] either because users are too engaged in their tasks to consider learning hotkeys [15] or because they do not have the means to estimate the potential benefits of switching to hotkey [48, 72]. Commands like 'copy' and 'find' are relatively easy to learn. They leverage a mnemonic mapping that maps the command to a hotkey combining a single modifier press Ctrl with the first letter of the command name (respectively C and F). On the other hand, a large vocabulary of commands means that multiple command names may share the same first letter, making it impossible to always use this mnemonic mapping strategy. This is why some hotkeys rely on combinations of modifier keys (typically Shift ⇧ or Alt alongside the main modifier)[3], or use a key that is not the first letter of the command, making hotkey learning more challenging.

In a 2011 paper, Scarr et al. [69] characterised intermodal performance improvement, that is, how performance evolves as users switch from one modality to another that can offer a higher performance ceiling, which is typically the case when switching from pointer-based to hotkey-based command selection. More interestingly, they advocate that a performance dip can be expected when switching from the first modality to the second, thus making users even less likely to adopt hotkeys. Previous research explored several approaches to foster and alleviate hotkey learning. For instance, Grossman et al. and Krisler and Alterman demonstrated that dedicated feedback (emphasising the hotkey after a command selection in the menu) and cost (imposing additional steps or delays) encourage hotkey adoption [29, 38]. While providing positive results, the cost-driven approach penalises regular user interaction. Moreover, users still have to memorise the hotkey in order to be able to use it.

Various solutions were explored to remove the necessity of having to memorise a hotkey before being able to use it. Typically, hardware-based solutions that rely on video projection [13] or LED displays [74] were used to display the commands' icons directly on the key used to activate it. Other solutions that do not require specific hardware were also proposed. For instance, ExposeHK [47] reveals all first-level menus of a menu bar at once or overlays toolbar items with the associated hotkeys when the modifier key is pressed, thus making it possible to select a hotkey without having to memorise it beforehand. Subsequently, there was a proposal of an alternative solution named IconHK [24] that directly blends visual cues to convey keyboard shortcut information using toolbar icons. Both ExposeHK and IconHK display the keyboard shortcut in place, where the corresponding command is displayed in the GUI. A different approach can be found in the Chrome OS [4] and KeyMap [45] that display command labels directly on an on-screen keyboard, thus leveraging Norman's principle of natural mapping [54]. Strategies like ExposeHK, IconHK, and KeyMap rely on spatially stable interfaces, thus capitalising on users prior knowledge of the interface, shortcut mapping and key

[3] see Apple macOS Human Interface Guidelines https://developer.apple.com/design/human-interface-guidelines/macos/user-interaction/keyboard/
[4] https://www.cnet.com/how-to/15-essential-chrome-os-keyboard-shortcuts/

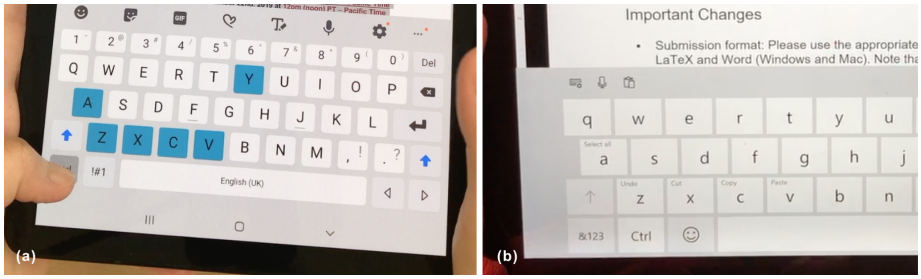

Fig. 1. Existing Implementations of Soft Keyboard Hotkeys on (a) Samsung Galaxy S and (b) Microsoft Surface. Images used with permission from Fennedy et al. [21].

locations to make rapid decisions instead of a slower visual search [17, 68]. These confirm the importance of spatial memory [62, 67, 70] in providing a foundation for designing an efficient command selection mechanism.

Our study is the first attempt to validate the learning benefits of using a keyboard layout for command selections on touch-based devices like smartphones and tablets.

2.3 Soft Keyboard Hotkeys

Since the adoption of desktop computers as office workstations, keyboard shortcuts have been inherently associated with physical keyboards. This might explain why the first generations of now mainstream touch-based operating systems (notably, Apple iOS and Google Android) did not activate keyboard shortcuts through the built-in software keyboards. However, multitouch devices like smartphones and tablets are equipped with soft keyboards, which offer great opportunities as they can dynamically render contextual details [20] while switching between text-entry and command-selection mode. As a result, recent versions of Samsung Galaxy [66] and Microsoft Surface [59] tablets offer keyboard shortcuts that can be activated by first holding/tapping on the Ctrl modifier key before hitting the key corresponding to the desired command (Figure 1). For phones and tablets, there was a third-party keyboard app called Swype [57] that invoked hotkeys using swiping gestures, but this service was discontinued [36] in 2018.

Upon activating the command-selection mode, Microsoft displays the command name above the letter of each hotkey. In contrast, Samsung only highlights the hotkeys in blue colour without displaying the corresponding command name, requiring users to know the hotkey beforehand. Interestingly, the visual feedback on Microsoft tablets is only applied to a subset of all the available commands (like 'Cut', 'Copy', and 'Paste'), while other commands can also be activated with a hotkey. For instance, keys like B, U, I are intuitively associated with bolding, underlining and italicising texts, but Samsung and Microsoft apply no visual effect to these other frequently triggered keys.

What immediately strikes with these implementations is that there are neither consistent input methods to activate the command-selection mode nor consistent visual representation of the keys when the command-selection mode has been activated. A recent study demonstrated that the input method 'Once' (which relies on two sequential taps) is faster than holding (User Maintained) or swiping (Swipe) the modifier key across device configurations and mobility conditions [21]. However, this work only focuses on input methods and has yet to investigate the visual representation, discoverability or retention performance. Hence, we are motivated to contribute a more in-depth understanding of the role of visual design in affecting the discoverability, performance, and user preference of SoftCuts.

2.4 Supporting Discoverability of Commands

Most usability guidelines [19, 52, 53, 71] promote the need to better support novice users when interacting with GUIs. Without prior knowledge, novices are often at a disadvantage initially as they rely on visual search or trial and error. This challenge is made worse when feature-rich applications need to support a large vocabulary of commands but over a limited screen real estate offered by today's smartphones. One solution is to nest these commands in a menu hierarchy but traversing through it still requires multiple steps, which are slow and not ideal for frequently selected commands. Alternatives come in the form of gestures, but despite its efficiency, they are considered "a step backward in usability" [56] because aside from the unnatural mapping [55], it is hard to discover [49, 64], hence the reason for its relatively low adoption rate [8, 64]. Feedforward strategies have been adopted by Marking Menu [39] and OctoPocus [11] to combat these discoverability issues by informing users what stroke gestures can be drawn and what the results will be. However, these techniques are rarely implemented in commercial devices. Moreover, solutions like Marking Menus suffer from a screen space constraint [9, 77] since it can be used mostly when the gesture begins in the central area of the display. On the other hand, using a keyboard offers a consistent and promising alternative feedforward approach that enables users to discover what features exist and how to find them regardless of what device is being used, while still supporting a relatively high number of commands.

2.5 Leveraging Prior Knowledge Across Devices

A performance dip [69] can also be expected in multi-device contexts, where differences in screen sizes and interaction paradigm [1] may end up having users re-learn skills as a novice. Therefore it has become more important than ever to consider designs that support cross-device learnability and eventually promote inter-usability [18]. Lafreniere et al. [40] also highlighted a post-training persistence to use expert method despite the absence of high-performance requirement. Hence, leveraging skill transfer [73] is one strategy that we can adopt by minimising the difference between the production rules [2] in the two contexts. For instance, a study has shown that touch-typing skill with a QWERTY layout keyboard could transfer to several variant keyboard designs [28]. It can potentially be generalised to keyboard shortcut skills when once learned, and automaticity reached [35], are also transferable from hard to soft keyboard, or vice versa. Our study remains the first attempt to assess the impact of leveraging prior known mapping (*i.e.* text-editing keyboard shortcuts) for touch-based command selection performance and recall.

3 MOTIVATION FOR STUDYING SOFTCUTS

This work aims to assess whether SoftCuts could be used as a general-purpose command selection mechanism across touch-based devices. While SoftCuts have been implemented on some commercial products, very little research was specifically conducted on them. To the best of our knowledge, the paper from Fennedy et al. [21] remains the only work specifically focused on SoftCuts to this date. Their work, however, focuses on comparing the performance and adoption of different input methods under different size (phone or tablet) and handling (one-handed or two-handed) conditions. While these results are of obvious importance for command selection, we believe that for SoftCuts to be widely adopted and used on touch-based devices, the technique needs to demonstrate additional characteristics that we discuss below.

3.1 Visual Representation

As pointed in previous work [21], implementations of SoftCuts on Microsoft Surface and Samsung Tabs use different visual layouts to show commands on the keys of the soft-keyboard. Samsung simply highlights the key associated with

a command, while Microsoft Surface provides the name of the command in addition to the letter name. In addition, Fennedy et al. [21] used a representation combining an icon and the name of the command of the key. We thus first decided to investigate the visual representations and understand which representation is preferred. We believe that only showing the letter will make it hard for first-time users to guess what the technique does and find a command they want to use. However, it is unclear which of the command name or icon-based designs should be preferred. Therefore, we formulate our first hypothesis:

H1: Users will not prefer the design with only letter being displayed.

3.2 Discoverability

Any general-purpose command selection technique should (ideally) be available in any application. More importantly, the mechanism to activate the technique should be easy to discover so that users would start using the technique and get used to it across different applications and devices. The current implementation of SoftCuts adds two small modifier keys on the soft keyboard that enable commands to be revealed after pressing one of them. In a scenario where the keyboard is shown on the screen, users would need to notice one of these modifier keys, press it and guess (from the visual representation) which commands they may invoke from there. Both commercial implementations use Ctrl as a familiar modifier key that is also used on physical keyboards. We found the trade-off between familiarity and saliency interesting to investigate. For keyboard-less scenarios, Fennedy et al. [21] suggested displaying the modifier key(s) on screen to increase its saliency. While this does consume space, the key itself is relatively small, and having only this key displayed on screen should make the technique easier to discover. The familiarity vs. saliency trade-off also comes into play: in that case, a salient familiar modifier key might lead the users towards tapping on it, revealing the technique, while an unfamiliar symbol could be confused for a floating action button with an unknown feature. We thus formulate our second hypothesis:

H2a: SoftCuts are easier to discover if the modifier key shows a familiar symbol or label (i.e. Ctrl or ⌘).

H2b: SoftCuts are easier to discover if the modifier key is presented without the remaining keyboard (*i.e.*keyboardless scenarios).

3.3 Prior Knowledge and Retention

One advantage of SoftCuts over existing command selection techniques is that it is similar to desktop hotkeys. As such, it is possible to use on touch-based devices the same command/key mapping that on desktop computers, with which users may be familiar and capitalise. This would provide a strong advantage to users who are familiar with these existing mappings, improve their selection performance, and help long-term retention. However, not every mobile application has a corresponding desktop application from which it could get its mapping. SoftCuts will still need users to learn new command/key mappings for these applications. Thus, we are interested in finding out how easy it is for users to learn SoftCuts, and how prior knowledge helps them in this learning process. This leads us to our third hypothesis:

H3a: Participants are faster when selecting commands from a mapping they are familiar with.

H3b: Participants are more accurate when selecting commands from a mapping they are familiar with.

H3c: Participants have a higher retention rate for mappings they are already familiar with as compared to newly learned mappings.

3.4 Performance

FastTap is one of the few command selection techniques that has been proposed on a variety of touch-based devices [31, 41]. It is a technique initially introduced for phablets and tablets that uses thumb-and-finger chords to support command selection from a spatially stable grid-based overlay interface. FastTap differs from SoftCuts in various aspects. First, it is a rehearsal-based interface (RBI) in which the command overlay is hidden by default and is shown only when the user holds their thumb on an activation button for a given duration (e.g. 250 ms). Hence, users have to know where a command is located in the grid and select it with a 'chorded' tap using the thumb and forefinger to select the command before the grid appears. Second, FastTap relies on a full-screen grid interface that displays larger buttons than the keys of a soft-keyboard.

SoftCuts are not intended to be an RBI because soft keyboard layouts update the keys instantaneously (and not after a delay) when some specific keys are pressed (for instance, the Shift ⇧ key). Then, SoftCuts use the soft keyboard layout, which occupies a significantly smaller portion of the screen than a full-screen grid, at the cost of smaller actionable buttons. However, this keyboard layout leverages familiarity with hotkeys, possibly allowing first-time users to perform command selection efficiently with minimum training. It is also easier to make commands spatially stable across applications as users are less likely to be surprised by a deactivated key that is not associated with a command, thus not requiring a re-flow of the commands that may disrupt users and impact their performance [68].

Performance remains an essential point for adopting a technique, as if a technique is too slow, it may simply never be used [72]. However, directly comparing the performance of SoftCuts and FastTap is of little interest and would be highly unfair, given how different these techniques are by nature. First, there is no prior knowledge that users can leverage as easily and clearly as existing hotkeys. Second, FastTap would suffer from a delay penalty for some selections, especially since some studies have shown that users would never stop selecting after this delay [26, 30] and make errors due to recall issues. However, what is of higher interest is how much SoftCuts compare to a full-screen grid-based layout command mechanism that displays larger targets to quantify the magnitude of the impact that the choice of relying on a keyboard layout has on performance. We believe that since FastTap offers larger areas for each command, selection time should be shorter and accuracy higher than SoftCuts. We can then derive our fourth hypothesis:
H4a: Selection time is lower on a grid layout as compared to a keyboard layout on tablets and phones.
H4b: Selection accuracy is higher on a grid layout as compared to a keyboard layout on tablets and phones.

Fig. 2. Design variations for SoftCuts. (a) We modified the existing iOS keyboard by adding two command keys at the bottom row. (b) Design 1: Available shortcut keys retain their default appearance, while non-shortcut keys are greyed and inactive. (b) Design 2: In addition to Design 1, the name of each command is printed at the bottom of respective keys. Design 3: Similar to Design 2, but substituting the key labels with icons instead.

4 STUDY 1: VISUAL REPRESENTATION OF COMMANDS

As illustrated by current commercial implementations of SoftCuts, there is no broad consensus on how to present hotkeys on a soft-keyboard. Indeed, the dynamic nature of soft keyboards being rendered on screen raises the question of how SoftCuts could be best presented to the user. In order to answer this question, we conducted semi-structured interviews and subjective ratings to evaluate the usability of three design variants (Figure 2). While we acknowledge that there are numerous design alternatives to represent the commands, we chose to focus on the following variants currently adopted by commercial systems [59, 66] or proposed by a previous study [21].

4.1 Design Variants

4.1.1 D1: Letter only. As illustrated in Figure 2b, this design deactivates keys that do not provide any shortcut functionality and focus the users' attention on active keys that offer shortcut functionality. D1 corresponds to the one currently being implemented on a Samsung tablet [66]. This design has the least amount of change from a standard soft keyboard among all the designs. It requires users to have prior knowledge of the available commands because the system does not provide mapping information between keys and commands. Novice users would be at a disadvantage because they would first have to guess the associated function and learn through trial and error or attempt to transfer skill learned from a different platform, resulting in a potential mapping mismatch.

4.1.2 D2: Letter and name. In this design, the command name is added at the bottom of each key, right below its corresponding letter. D2 is similar to the soft keyboard hotkey implementation of Microsoft Surface tablets [59] while only differing in command name position and saliency (Figure 2c). This design facilitates direct mappings between each letter and its corresponding command output. Over repeated triggers, D2 may help long-term retention of explicit mapping printed on each key and potentially support skill transfer from soft keyboards to physical keyboards.

4.1.3 D3: Icon and name. Figure 2d shows how this design hides the letter from the key to only show the command name and a corresponding pictorial representation (icon). D3 has only been used by Fennedy et al. [21], while some physical keyboards [13, 74] have attempted to pursue a similar direction. This design uses icons as a visual cue for guiding visual search [10] and conveying command meaning [24].

4.2 Procedure

We conducted the study through a video conferencing platform by starting with participants digitally signing a consent form and then downloading the prototypes on their phone. The prototype offered two simulated apps: "Notes" for note-taking and "Safari" for web-browsing tasks. For the first app, we instructed participants to complete tasks using one of the design variants before they rated their experience in terms of preference, ease of use, perceived speed, perceived accuracy, usefulness, comfort, and visual appeal using a 7-point Likert Scale. Also, participants were invited to an optional open-ended discussion to suggest any feedback and ask the experimenter any question. They would repeat the tasks, rating, and discussion phase with the second and third design variant. We then finally asked for their overall design preference (D1, D2, or D3) for that particular app. By the second (final) app, participants were expected to be familiar with the rationale behind SoftCuts already. We wanted to see how their comments would change as we switched the app in which they had to complete similar tasks. The session ended with asking participants which keyboard shortcuts they were most familiar with.

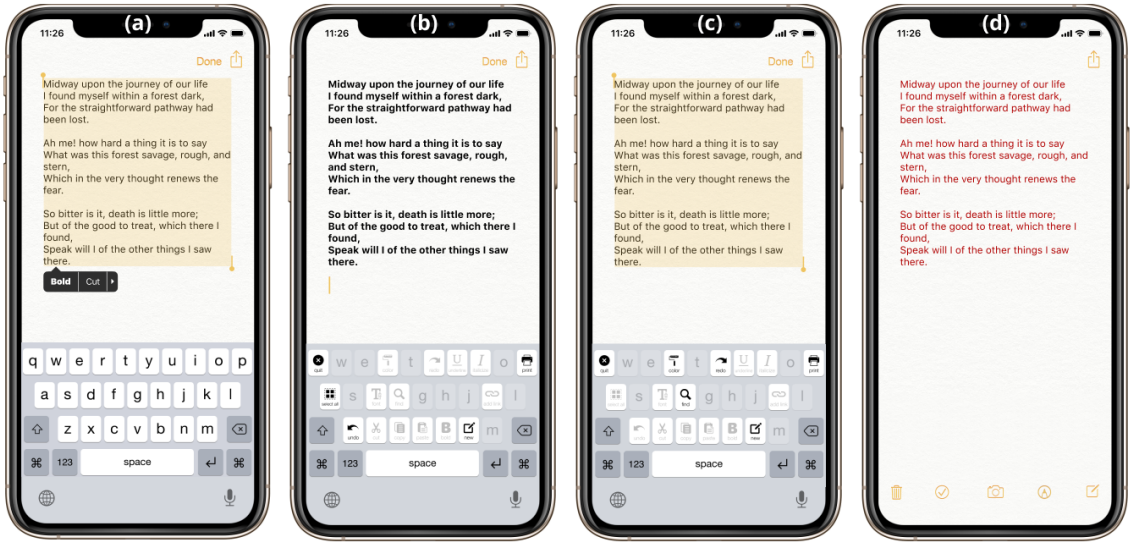

Fig. 3. A demonstration of tasks in a note-taking app that may be done using existing mechanisms or SoftCuts invocations to complete. Participants had to paste text, (a) select all the pasted text, (b) bold the selected text, (c) undo the bold formatting then select all the text again, and finally (d) colour the selected text red. Step (c) and (d) could only be achieved through SoftCuts.

4.3 Task

We slightly altered the three design variants to accommodate the limited functionality of both the Notes and Browser app prototype[5]. As seen in Figure 3b, 3c, and 4b, the white keys contain either black or light grey font. The black font represents functional commands, while light grey represents disabled commands.

4.3.1 Note-taking App (Keyboard Task). This app offers 16 shortcuts and features a scenario with a visual on-screen keyboard (Figure 3). We asked participants to assume that text had been previously copied and stored in the OS clipboard. Then participants had to perform: 1) paste text, 2) select all pasted text, 3) make it bold, 4) undo the bold command, 5) select the entire text again, and 6) colour the text in red.

Commands 1, 2, 3, and 5 could be invoked either using the default menus or via SoftCuts. Commands 4 and 6 [6] could only be invoked using SoftCuts located in the Z and R keys respectively. Keys for SoftCuts were chosen either because they correspond to the actual keyboard shortcut on the desktop for this command (*e.g.* V , A , Z , F , D) or because it was the first letter of the command name (*e.g.* R for red colour). We intentionally chose to rely on commands that could only be activated via SoftCuts to encourage participants to explore beyond their familiar menu-based approach and thus reveal the tested SoftCuts visual design as they would when interacting with their device.

4.3.2 Web-browsing App (Keyboardless Task). This app offered nine shortcuts keys and featured a scenario without any keyboard initially, with only the (modifier) ⌘ key presented (Figure 4). When using this app, participants had to activate: 1) "search on page" command and 2) bookmark command. These two functions could only be activated via

[5]Please refer to our accompanying video for demonstration of all the tasks specified in each app.
[6]Note that the undo command on iOS can also be triggered by shaking the device, which we did not reproduce.

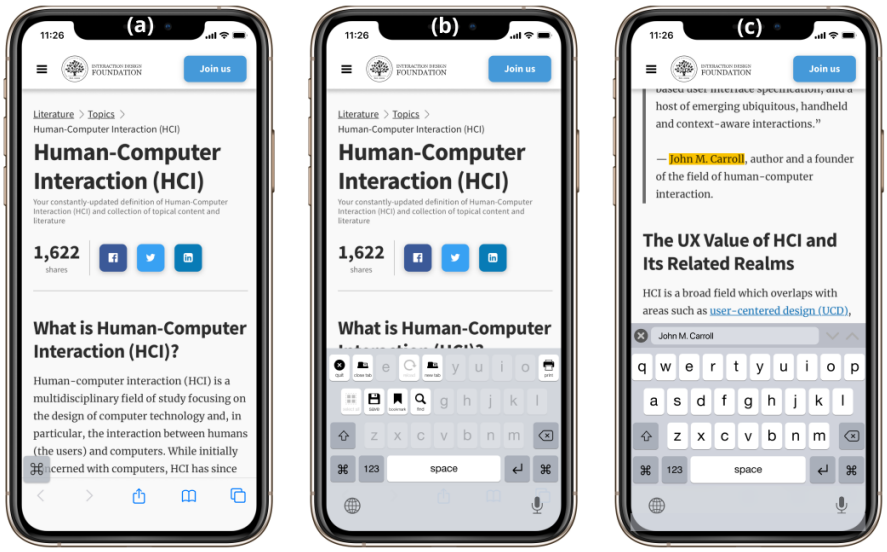

Fig. 4. A demonstration of the search task in web-browsing app that can only be completed using SoftCuts. Participants had to (a) press the command key at the bottom left, then (b) the soft keyboard would appear with active keys that correspond to available keyboard shortcuts, and when they would press the F key, (c) the search bar would appear above the keyboard with a preset text and the page will animate a scrolling down effect until the text has been identified.

SoftCuts, and no native approach was implemented. We chose the F key for find and the D key for bookmark as we wanted to preserve the mapping that already exists on desktop computers.

4.4 Participants and Apparatus

We recruited 12 participants (10 female, 2 male and all right-handed), aged 20 to 23 (M=21.9, SD=1.0) from the university community. The recruitment was limited to iPhone users to ensure familiarity with iOS user interface design and the ⌘ key. The majority of the participants reported that they used keyboard shortcuts either "often" or "always" on a 1-5 Likert Scale (1: never, 5: always, M=4.1, SD=1.0). We designed our high-fidelity note-taking and web-browsing app prototypes using Figma [22] and hosted them via Maze [50]. Participants used their iPhones (models include iPhone X, XR, XS and 11) to run the prototype and a computer with access to WiFi to facilitate the online interview and subjective rating. Each participant received the equivalent of US$ 5.5 reimbursement for their 45 minutes participation.

4.5 Design

We used a within-subject design with DESIGN as a primary independent variable and APP as a second independent variable. DESIGN has 3 levels { D1, D2, and D3 } while APP has 2 levels { NOTES and BROWSER }. The order of presentation of the two apps and three design variants were counterbalanced using Latin Square to avoid any potential ordering effect. In terms of dependent variables, we measured subjective feedback using a 1-7 Likert (1: strongly disagree, 7: strongly agree) on their overall opinion between design variant D1, D2 and D3 in terms of *like*, *ease of use*, *speed*, *accuracy*, *comfort*, *usefulness*, and visual *appeal*. The experiment lasted for around 45 to 60 minutes. In summary, we recorded 12 participants × 2 apps × 3 designs × 7 rating questions = 504 ratings in total.

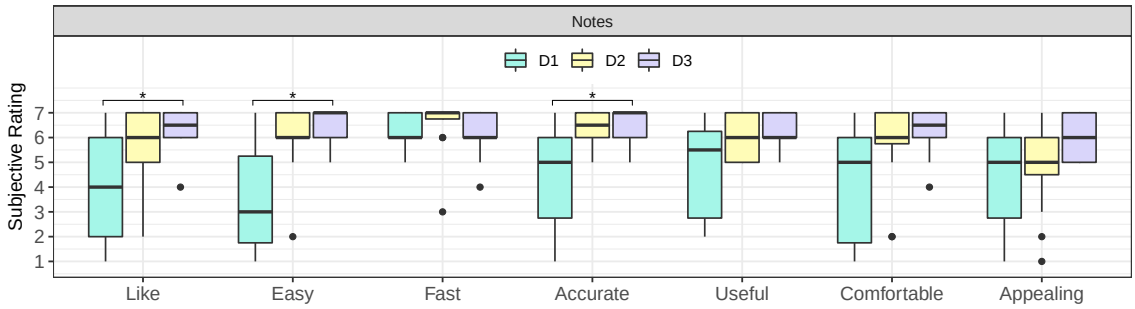

Fig. 5. Subjective rating for each DESIGN of NOTES app (1: Strongly Disagree, 7: Strongly Agree).

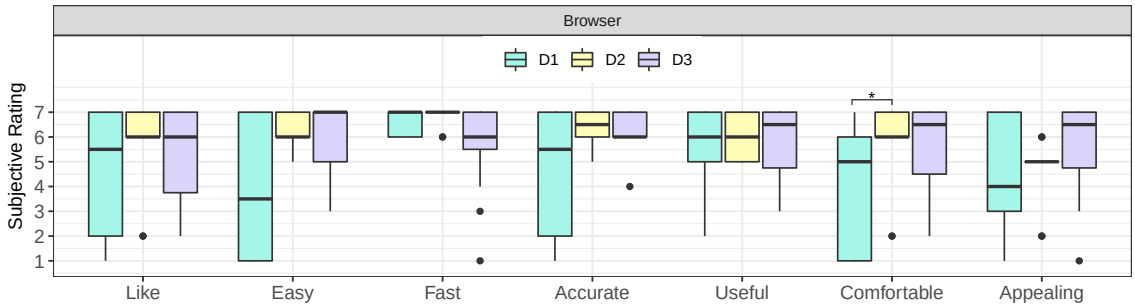

Fig. 6. Subjective rating for each DESIGN of BROWSER app (1: Strongly Disagree, 7: Strongly Agree).

4.6 Results

For quantitative data analysis from Likert scale rating, we used Friedman tests (non-parametric) and used Wilcoxon signed-rank tests with Bonferroni corrections for post hoc comparisons.

4.6.1 Notes (Keyboard Scenario). The majority of participants liked D3, found it easier and more accurate to use than D1 in NOTES app (Figure 5). There were significant main effects of DESIGN on *like* ($\chi^2(2, N = 12) = 8.49$, $p = .014$), *ease of use* ($\chi^2(2, N = 12) = 8.60$, $p = .014$), *accuracy* ($\chi^2(2, N = 12) = 10.61$, $p < .01$), *comfort* ($\chi^2(2, N = 12) = 7.82$, $p = .020$), and *appeal* ($\chi^2(2, N = 12) = 8.16$, $p = .017$). Pairwise comparisons revealed that 8 out of 12 participants liked ($p = .044$) D3 (median=6.5) more than D1 (median=4.0). 9 out of 12 participants found D3 (median=7.0) easier to use ($p = .027$) than D1 (median=3.0). 9 out of 12 participants found D3 (median=6.5) more accurate ($p = .032$) than D1 (median=5.0). Scores for D1 tended to be lower than D3 but we did not find any significant differences between D1 and D2 or between D2 and D3 either. **H1 (novice users will not prefer D1)** is thus supported.

4.6.2 Browser (Keyboardless Scenario). Participants found D2 more comfortable to use than D1 in BROWSER app (Figure 6). There were significant main effects of DESIGN on *speed* ($\chi^2(2, N = 12) = 6.93$, $p = .031$), *accuracy* ($\chi^2(2, N = 12) = 6.06$, $p = .048$), *comfort* ($\chi^2(2, N = 12) = 8.19$, $p = .017$), and *appeal* ($\chi^2(2, N = 12) = 7.17$, $p = .028$). Pairwise comparisons revealed that 8 out of 12 participants found D2 (median=6.0) more comfortable to use ($p = .039$) than D1 (median=5.0). There were no other differences between individual designs.

4.7 Discussion

4.7.1 Strong preference for D3. From our quantitative results in Notes app, D3 was preferred over D1 in terms of likeability ($\Delta median = 2.5$), ease of use ($\Delta median = 4.0$), and accuracy ($\Delta median = 1.5$). At least 75% of participants overall chose D3 over D2 for both NOTES (9 vs. 3) and BROWSER (10 vs. 1) app. Participants explained that *"it is easier and faster to infer what each key represents from icons as compared to reading the name labels at the bottom"* [P1] (similar opinions from 7 other participants). This is especially true when the icons are consistent with those used in existing commercial apps like Microsoft Word that they are already familiar with (mentioned by P3 and P5). Therefore, we would recommend D3 as a suitable design for SoftCuts.

4.7.2 D2 as the next best alternative. Despite a preference for D3, we were curious as to why some participants would choose D2 for NOTES (3 persons) and BROWSER (1 person) app. One reason mentioned was that with the already limited screen space available for each key, letters in D2 on average consume less space than the icons in D3, resulting in *"a cleaner, minimalist"* [P2] and *"less cluttered"* [P10] presentation of commands in D2. Another reason could be that icons can be misinterpreted [10], especially with commands that do not have a standardised visual representation. By using both letter and command name on each key, D2 has the potential to build stronger shortcut mappings than D3 when new contexts may involve keyboards that do not offer any cues (*e.g.* physical keyboard or Samsung tablet's soft keyboard [66]). Therefore, it is a worthy consideration to offer individual user customisation where they can *"flexibly change preferred design between D2 and D3"* [P6,10,12], depending on their level of familiarity.

4.7.3 SoftCuts may facilitate knowledge transfer and expansion. Our assumption on leveraging prior knowledge is partially supported when participants affirmed: *"it is pretty cool that [SoftCuts] mirror what you do on a physical keyboard"* [P1]. This suggests a minimal performance dip [16], *"as long as one has been using physical keyboard shortcuts for a long time, it would be very easy and not take too long to get used to SoftCuts on mobile devices"* [P5]. Beyond intermodal performance improvement, SoftCuts also facilitates command browsing because *"it lets me know what other commands are available for future invocations"* [P3,7,9] and as such may increase participants' mapping vocabulary through implicit learning and frequent exposure. For instance, despite not being tasked to invoke a redo command, P7 expressed his discovery through SoftCuts that ⌘ + Y shortcut can be used to trigger the redo command next time. Therefore, these observations suggest how robust SoftCuts could support and even grow expert performance.

4.7.4 Limitations. Design combinations like D2 and D3 face space constraints when accommodating long command names like "Paste and Match Style". Decreasing the font to fit the very small keys on a smartphone will only worsen the accessibility as the current font size may have already posed a challenge for an older population. However, visualisation strategies could be explored in future studies. For instance, when users are holding a particular key with a finger, a bubble callout [75] can be used to magnify the small font or offset the presentation of long command names to the available space above the keyboard. Another strategy is to adopt scrolling animation. Alternatively, designers could also establish a standardised icon to represent these commands such that the icon alone is sufficient to communicate the function of the key. We also acknowledge that our study with participants of 20-23 years old and have used hotkeys frequently may be limited to generalise the results. Additional studies are needed to confirm if a similar design preference could be observed for novices or someone who has not used hotkeys at all.

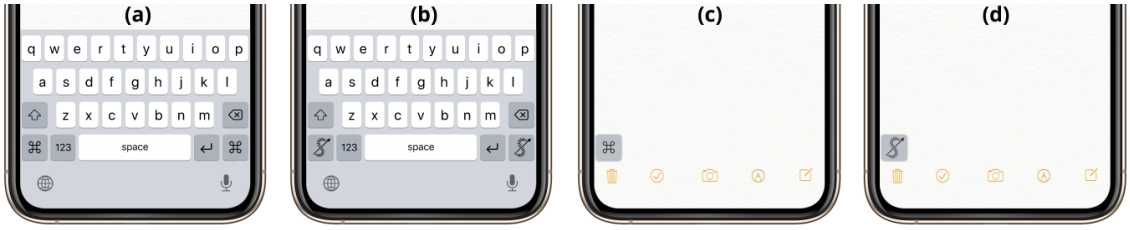

Fig. 7. The bottom section of starting screen while varying both FAMILIARITY and SALIENCY: (a) Keyboard (KB) + Command as modifier, (b) KB + Custom modifier, (c) Keyboardless (noKB) + Command, and (d) noKB + Custom.

5 STUDY 2: DISCOVERABILITY OF SOFTCUTS

5.1 Study Rationale

Previous work on SoftCuts [21] postulated that users' familiarity with desktop keyboard shortcuts would make SoftCuts easy to discover but highlighted that future work needs to confirm this assumption. In this second study, our goal was to assess whether users spontaneously discover that they can use SoftCuts, that is if users would start to use SoftCuts without being explicitly introduced to them prior. The discovery of an interaction technique is a binary measure. As such, it can only happen once for each participant, and repeated measures are impossible. Thus, it requires a large number of participants to generate insightful analysis, which is why we chose to recruit participants from Amazon Mechanical Turk. This study was conducted by deploying a prototype of our system on Amazon Mechanical Turk and invited crowdworkers to perform a series of tasks in this experiment. Some tasks required users to select commands that could be activated either using SoftCuts or using the native menu/toolbar. Other tasks required users to select commands that could only be activated using SoftCuts, thus requiring participants to discover the technique.

We were interested in finding out if having the modifier key alone on screen would give a more explicit hint to participants and allow them to discover SoftCuts more easily. In keyboardless (noKB) conditions, participants would see a single modifier key on the bottom left of their screen (Figure 7c and d). In keyboard (KB) conditions, participants would see the whole on-screen keyboard with the modifier key located at either bottom left or bottom right corner of the said keyboard (Figure 7a and b). These conditions vary the saliency of the modifier key. We also considered two different visual designs for the character used on the modifier key: a familiar ⌘ / Ctrl label and a custom one . The input method used for SoftCuts in this study is the "Once" technique, where commands are selected by sequentially tapping on the modifier key and then on the command key. Based on the previous study results (Section 4), we decided that when the modifier key is pressed, keys that could activate commands will display the icon and the name of the corresponding command (*i.e.* D3).

5.2 Missions and Tasks

We designed two missions for participants to complete, consisting of tasks replicated from the previous study using the Notes app prototype (see our accompanying video for details). Mission1 features three tasks: pasting, selecting, and bolding texts. A unique design of Mission1 is that all tasks can be completed using either the default menus or SoftCuts. Mission2 also features three tasks: undoing commands, selecting, and colouring texts. However, unlike Mission1, the first and last tasks are only doable using SoftCuts. As such, Mission2 can only be completed if the user has discovered SoftCuts. Each mission offers a 'Give up' button for participants who wish to end their tasks early, but we inform participants at the beginning of the study that their rewards are based on their performance.

5.3 Procedure

After participants signed the consent form and answered demographic questions, we assigned them to one of the four conditions, which are combinations of saliency (KB vs.noKB) and familiarity (⌘ / Ctrl vs. ⌘) of modifier keys. The four conditions are KB+Cmd/Ctrl, KB+Custom, noKB+Cmd/Ctrl, and noKB+Custom. We asked participants which desktop OS they are most familiar with (macOS, Windows, or Linux). If they chose macOS, we would use the ⌘ label in the modifier keys for both KB+Cmd/Ctrl and noKB+Cmd/Ctrl conditions. Else, we would use the Ctrl label instead. In each condition, participants had to pass Mission1 first before proceeding to Mission2. After completing both missions, participants were introduced to the concept (the what, the how, and the why) of SoftCuts. We then asked them to rate their experience with SoftCuts using a 7-point Likert scale. To test participants' level of attention halfway through the Likert rating, we asked a trick question instructing the participant to choose a given option intentionally. If they chose otherwise, we would invalidate their submissions. Please refer to our supplementary document for more details about the questions we asked. While our prototype was using the D3 design variant of SoftCuts, we asked participants which one they would prefer if compared with D1 and D2, as featured in our first study (Section 4). To capture richer qualitative feedback, we had an open-ended section where participants could type any other thoughts or suggestions about SoftCuts.

5.4 Measuring Discoverability

For each mission, we count how many participants were able to discover SoftCuts under different conditions. We distinguish three main metrics to measure discoverability.

5.4.1 Spontaneous discovery rate. This metric measures discovery of SoftCuts when users are not required to use it to complete a task. Similar to [4, 27], it is computed as the ratio between the number of participants who used SoftCuts in Mission1 and the number of participants who actually started Mission1 (Eq. 1).

$$\text{Spontaneous discovery rate} = \frac{\text{\# of participants who used SoftCuts in Mission1}}{\text{\# of participants who started Mission1}} \tag{1}$$

5.4.2 Enforced discovery rate. This metric measures discovery of SoftCuts when users need it to complete a task. It is computed as the ratio between the number of participants who used SoftCuts only in Mission2 (but not in Mission1) and the number of participants who had not used SoftCuts in Mission1 and started Mission2. This is because 1) only upon completing Mission1 successfully will participants be allowed to proceed to Mission2 and 2) discovering SoftCuts in Mission1 means that participants already knew about SoftCuts when in Mission2 and hence did not need to 'discover' it again (Eq. 2).

$$\text{Enforced discovery rate} = \frac{\text{\# of participants who used SoftCuts only in Mission2}}{\text{\# of participants who started Mission2 but did not use SoftCuts in Mission1}} \tag{2}$$

5.4.3 Overall discovery rate. We measure the overall discovery rate of the study as the ratio between the number of participants who used SoftCuts at least once during the study and the total number of participants (Eq. 3). This is not a simple addition between spontaneous discovery rate and enforced discovery rate because there may be duplicates when the former overlaps with the latter, that is when participants who use SoftCuts in Mission1 may use it again in Mission2.

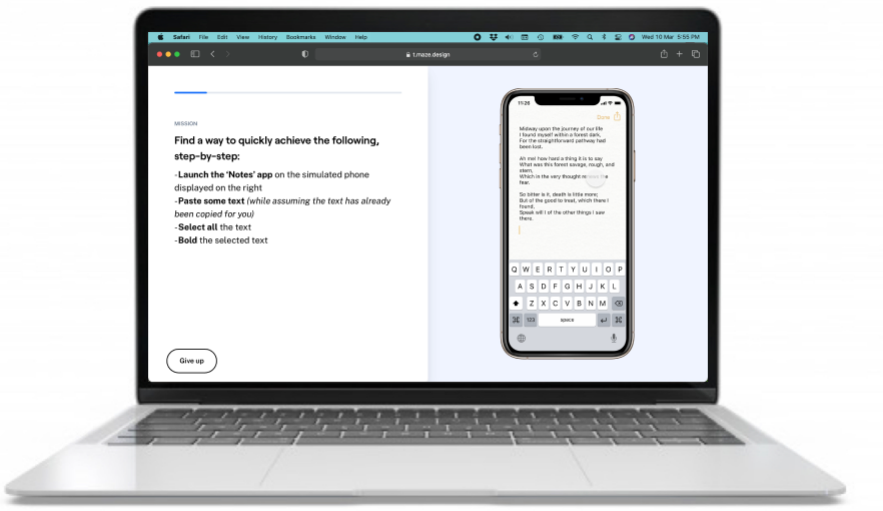

Fig. 8. A desktop version of the mobile prototype being deployed to each crowdworker.

$$\text{Overall discovery rate} = \frac{\text{\# of participants who used SoftCuts during the study}}{\text{Total \# of participants in the study}} \tag{3}$$

5.5 Participants and Apparatus

A total of 248 participants were recruited from the United States using Amazon Mechanical Turk. 160 of them (77 were male, 81 were female, 1 of them non-binary, and 1 preferred not to disclose) passed the attention test and were assigned to one of the 4 conditions evenly. Participants age ranged between 19 and 75 years old (M=36.3, SD=12.2). We required participants to have at least 95% approval rate. We also granted qualifications to workers to ensure that they could only complete the experiment once. Given that our experimental interface was based on the iOS Notes application, we also required participants to be familiar with iPhones. For smartphone, only 2 (1.3%) of the participants had 0 to 2 years of experience, 15 (9.4%) had 2 to 5 years, 121 (75.6%) had 5 to 15 years, and 22 (13.8%) had more than 15 years. For computer, 10 (6.3%) of them had 2 to 5 years of experience, 58 (36.3%) had 5 to 15 years, and 92 (57.5%) had more than 15 years. The majority reported that they used keyboard shortcuts either often or always on a range from 1 ("never") to 7 ("always") (M=5.0, SD=1.8). Giving up Mission1 results in reimbursement of US$0.26 for a maximum of 2 minutes of their participation. Giving up Mission2 results in reimbursement of US$0.65 for a maximum of 5 minutes of their participation. If participants did not give up at all, they would be reimbursed US$1.30 for a maximum of 10 minutes of their participation.

We deployed the same interactive note-taking app prototype (Fig. 3) used in the previous study on visual representation. Participants had to access the experimental interface from a desktop or a laptop computer using any web browser they desired (Fig. 8). Pointing and clicking interaction with the prototype can be facilitated using a mouse or a laptop's trackpad. We excluded mobile device interactions in the study for two reasons. First, the screen size of a tablet is significantly larger than that of a smartphone. It is challenging to ensure that all participants use a suitable model of similar screen size. Second, we cannot control the fact that some participants may choose to use their mobile devices

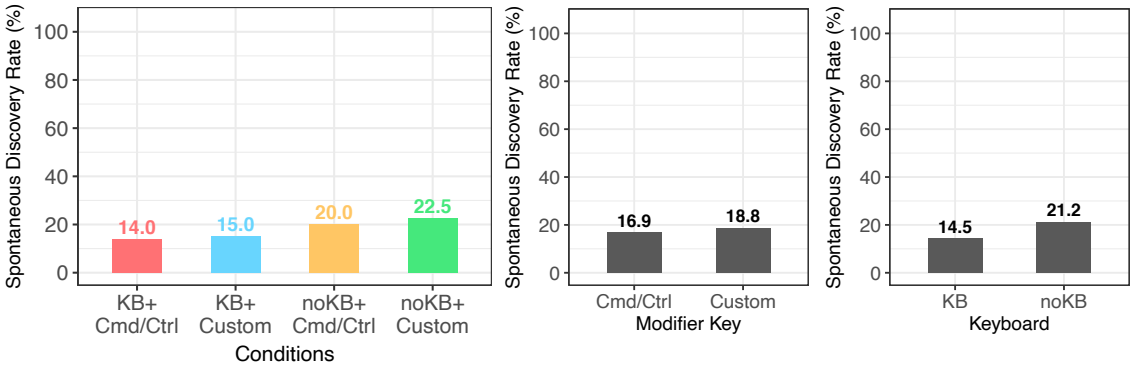

Fig. 9. *Spontaneous discovery rate* of SoftCuts (a) across all conditions, (b) aggregated by FAMILIARITY (*i.e.* type of modifier key used), and (c) aggregated by SALIENCY (*i.e.* presence of keyboard).

with an external keyboard, which supports keyboard shortcuts and potentially influences our results. In addition, we are primarily investigating behaviour differences instead of performance. Therefore, our strategy is to render the smartphone prototype on a desktop/laptop computer to maximise the quality of data collected while minimising variation in device ergonomics.

5.6 Design

We used a 2 × 2 between-subject design with two independent variables: FAMILIARITY { CMD/CTRL,CUSTOM } and SALIENCY { keyboard(KB), keyboardless(NOKB) }. We also have extraneous variables based on users' experience, namely hotkeys' USAGE FREQUENCY { 1-7 }, QUANTITY KNOWN/USED { 1-5, 6-10, 11-20, 21-30, >30 }, and USAGE DURATION { 0-2, 2-5, 5-15, >15 years } to better understand if users' familiarity with hotkeys do affect our results. In terms of dependent variables, we measured three *discovery rates* (*i.e.* spontaneous discovery rate, enforced discovery rate, and overall discovery rate). We also measured subjective feedback using a 1-7 Likert Scale (1: strongly disagree, 7: strongly agree) on their opinion using SoftCuts in terms of *usefulness*, *ease of use*, *discoverability*, *memorability* and *visual appeal*.

5.7 Results

A total of 9 (5.6%) participants failed Mission1, 59 (36.9%) passed Mission1 but failed Mission2, 92 (57.5%) passed both Mission1 and Mission2. Out of those who completed both missions, 2 (2.2%) preferred the alternative D1, 14 (15.2%) preferred D2, and the remaining 76 (82.6%) chose to stick with D3, which is the design they used in the study. We distinguished between mission success rate and SoftCuts discovery rate because the participant might have failed a given mission but still managed to activate a command using SoftCuts during the failed mission. Then, we analysed *discovery rates* using pairwise Chi-squared tests with necessary Yates' continuity correction, followed by a Chi-squared test with Bonferroni corrections for post-hoc comparisons. For subjective rating, we used the Kruskal-Wallis test, followed by pairwise Mann-Whitney U tests with Bonferroni corrections for post hoc comparisons.

5.7.1 Discovery rates.

Spontaneous discovery rate. Overall, participants discovered SoftCuts spontaneously at a similarly low rate of 18.1% (29 out of 160). There was no main effect of FAMILIARITY or SALIENCY on *spontaneous discovery rate* ($\chi^2 = 1.39$, $p = .7$).

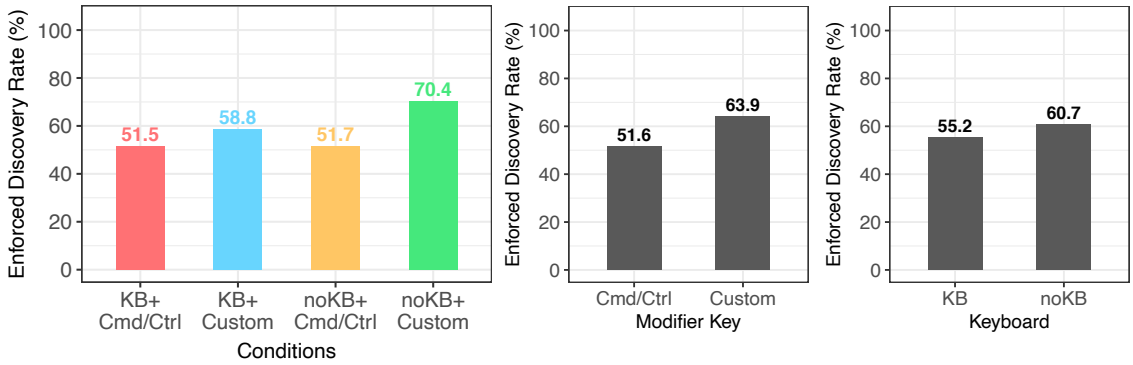

Fig. 10. *Enforced discovery rate* of SoftCuts (a) across all conditions, (b) aggregated by FAMILIARITY (*i.e.* type of modifier key used), and (c) aggregated by SALIENCY (*i.e.* presence of keyboard).

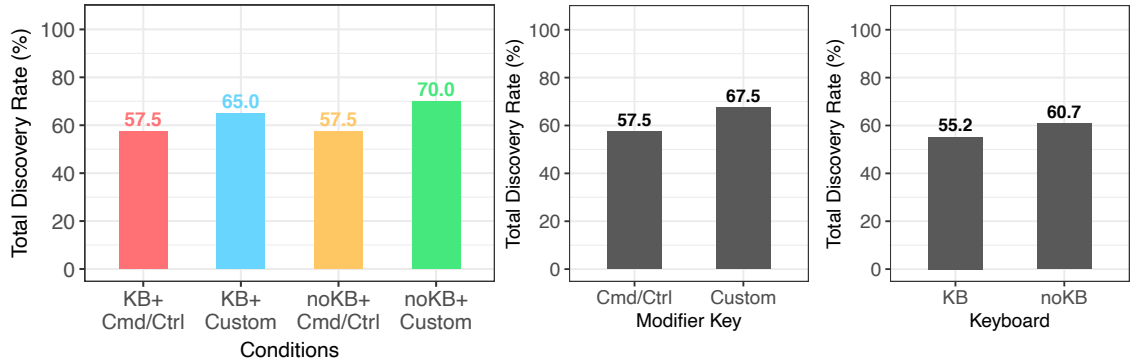

Fig. 11. *Overall discovery rate* of SoftCuts (a) across all conditions, (b) aggregated by FAMILIARITY (*i.e.* type of modifier key used), and (c) aggregated by SALIENCY (*i.e.* presence of keyboard).

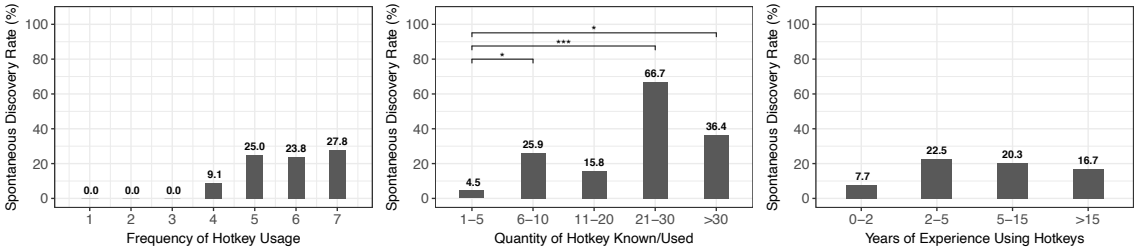

Fig. 12. *Spontaneous discovery rate* of SoftCuts across (a) frequency of hotkey usage, (b) quantity of hotkey known/used, and (c) years of experience using hotkeys.

However, more participants discovered SoftCuts when they declared having to know/use more hotkeys (Figure 12b). There was a significant main effect of QUANTITY of hotkey known/used ($\chi^2 = 22.60$, $p < .001$) on *spontaneous discovery rate*, with participants who declared knowing/using 6-10 (25.9%), 21-30 (66.7%), and >30 hotkeys (36.4%), achieving a higher rate (all $p < .05$) than those with only 1-5 hotkeys (4.5%).

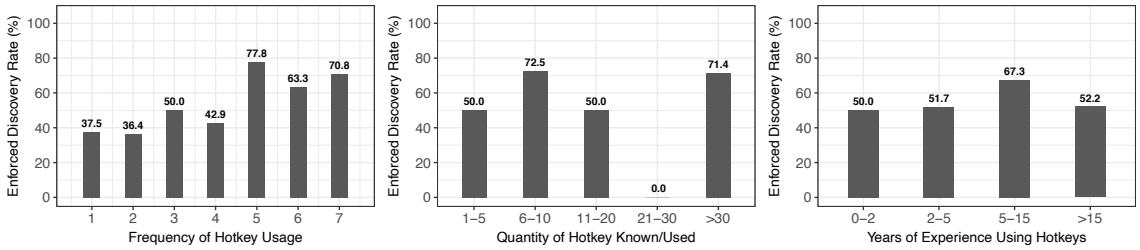

Fig. 13. *Enforced discovery rate* of SoftCuts across (a) frequency of hotkey usage, (b) quantity of hotkey known/used, and (c) years of experience using hotkeys.

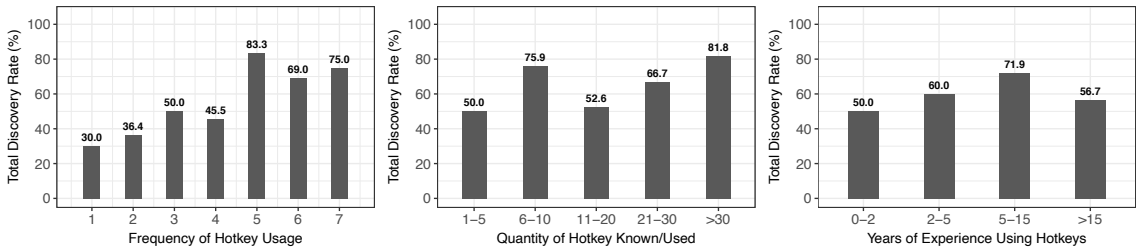

Fig. 14. *Overall discovery rate* of SoftCuts across (a) frequency of hotkey usage, (b) quantity of hotkey known/used, and (c) years of experience using hotkeys.

Enforced discovery rate. Overall, participants discovered SoftCuts at a higher rate of 57.7% (71 out of 123) when they are constrained to do so. We found a total of 123 participants who started Mission2 without using SoftCuts in Mission1. However, there was no main effect of FAMILIARITY, SALIENCY, hotkeys' USAGE FREQUENCY, QUANTITY KNOWN/USED, and USAGE DURATION on *enforced discovery rate* (all $p > .05$).

Overall discovery rate. Across both missions, participants discovered SoftCuts at a rate of 62.5% (100 out of 160). There was no main effect of FAMILIARITY, SALIENCY, hotkeys' USAGE FREQUENCY, QUANTITY KNOWN/USED, and USAGE DURATION on *overall discovery rate* (all $p > .05$).

Therefore, both **H2a (SoftCuts are easier to discover if the modifier key shows a familiar symbol)** and **H2b (SoftCuts are easier to discover if the remaining keyboard layout is absent)** are not supported. However, the familiarity of hotkeys, which we correlated with the number of hotkeys they already know on desktop computers, does help to boost discoverability.

5.7.2 Subjective Rating. Participants rated the usage of SoftCuts across both FAMILIARITY and SALIENCY similarly high (all medians >= 6), as shown in Figure 15. There was no main effect of FAMILIARITY or SALIENCY on any of the *subjective rating* (all $p > .05$). This suggests that SoftCuts are overall perceived positively across the conditions.

5.8 Discussion

5.8.1 Spontaneous vs. Enforced discovery rate. It is interesting to observe how the discovery rate tripled when a successful discovery of SoftCuts is the only pathway for users to complete the task, moving from a spontaneous discovery rate at 18.1% to an enforced discovery rate at 57.7%. The overall 62.5% discovery rate across both missions

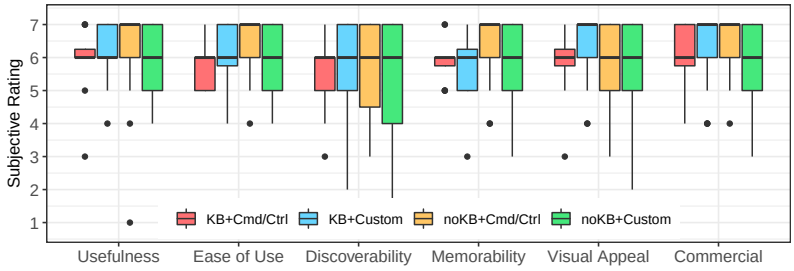

Fig. 15. Subjective rating for each CONDITION (1: Strongly Disagree, 7: Strongly Agree).

shows that participants could discover SoftCuts without any external help despite the short duration (less than 5 minutes) of interaction spent. However, we also consider ways on how to translate these positive trends to the real world. For instance, two participants (one from KB+CMD/CTRL and another from NOKB+CUSTOM condition) suggested "using a different colour" to increase the saliency of the modifier key. This suggestion may boost SoftCuts' discoverability because the trend from our discovery rates show that the more salient noKB conditions could perform better than its KB counterparts. We did not adopt a contrasting colour for the modifier key because it is important to preserve the aesthetic integrity and consistency of the user interface designs [7].

5.8.2 Hotkey usage familiarity drives discoverability regardless of modifier key label. Our quantitative results reveal that a familiar modifier key does not affect the discoverability rate of our participants. However, those who declared knowing/using a higher quantity of hotkey are more likely to discover SoftCuts. It is indeed surprising because we initially assumed that ⌘ / Ctrl are visually more familiar than a custom key, when in fact, it is the experience of using hotkeys that increase users inclination to discover by chance. We hypothesise that there is a potential difference between the visual and kinesthetic familiarity of users. This is because an expert accustomed to invoking hotkeys may not need to pay explicit attention to what label was printed on the modifier key, as long as they could remember its position to place their finger, which is more important in this case.

5.8.3 Qualitative benefits of using a familiar modifier key. There were 4 participants who used SoftCuts with the custom modifier key and suggested that its function remained unclear. Particularly, one of them commented, "I only clicked on the [ ] icon out of curiosity as to what it was. I think if I had knowledge of installing it for myself, I would have been familiar and known how to discover it." Another one of them from the KB+CUSTOM condition also suggested that "there should've been a short intro to what the symbol meant". However, promoting a feature during the onboarding process can do more harm than good, and hence we should consider avoiding it whenever possible [33]. Joyce [34] conducted a quantitative usability study that demonstrated how tutorials did not improve task completion time and were even perceived to make the task overly complicated. A solution worth trying to enhance the discoverability of SoftCuts is to use contextual tips, where users could be suggested to explore the more efficient SoftCuts after using the default menus for a long enough duration. Therefore, a familiar modifier key like ⌘ could at least communicate its meaning to users who have prior knowledge of hotkeys, as compared to whose function has yet to be known by anyone.

[7] https://developer.apple.com/design/human-interface-guidelines/ios/overview/themes/

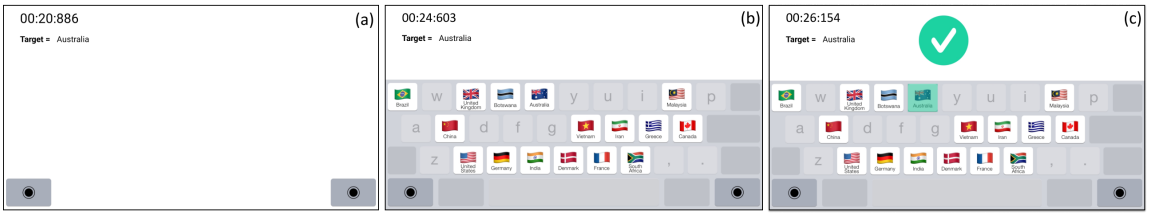

Fig. 16. Example of a trial in the experiment. (a) The participant was first presented with the stimulus "Australia", (b) they clicked on one of the left/right modifier keys at the bottom of the screen, then (c) they accurately selected "Australia" and received feedback. The abstract command set is shown on the (b) and (c) panels.

5.8.4 Overall positive feedback from crowdworkers. We were surprised to receive detailed feedback from numerous participants when they could easily choose to write a one-word answer for the open-ended final section of this survey. Particularly, 5 participants (1 from noKB+Cmd/Ctrl and 4 from noKB+Custom) similarly commended the compact keyboard layout used to organise the commands. For instance, one of them said, "I love how every command I need in the Notes app would be in one small little package on my keyboard. So simple and smart". In addition, 15 participants expressed interests to see SoftCuts integrated into their personal devices when they said, "I would really like to have SoftCuts on my iPhone since I don't know how to do the things on my notes the way i was able to do it on this demo" or "I would use it right away if it was implemented. Please implement this feature". Future work should consider launching SoftCuts as a third party keyboard plugin before each company like Apple and Google is ready to integrate SoftCuts into their respective ecosystem.

5.9 Limitation

Given the multidimensional nature [65] of crowdworkers' motivations, we also acknowledge that our results might demonstrate different results as compared to participants recruited in a lab study. For instance, in terms of the quantity of hotkey known/used, there is a possibility that crowdworkers were not accurate with their answers. Perhaps a more accurate relationship between familiarity and discoverability could be established if there was a way to guarantee a reliable assessment of crowdworkers' knowledge of hotkey mappings, which we are not aware of. Future studies should consider evaluating the discoverability of SoftCuts with participants in the lab to generalise our results further.

6 STUDY 3: LEARNABILITY OF SOFTCUTS

This study evaluates how easily users can learn command mappings and if prior knowledge of command mapping does help in this learning process.

6.1 Soft Keyboard Layout

The soft keyboard used in this study had the modifier keys located at its bottom-left and right corners (Figure 16 and 17). When either of the modifier keys was hit, the soft keyboard would be displayed instantly. Keys with associated commands would be rendered with both its icon and name. Keys without associated commands would be greyed out. This graphical representation is different from commercial solutions, highlighting the key (for Samsung) or revealing the command name at the top of the key (for Microsoft Surface). It was chosen based on the results from our first study and that icons act as an efficient visual cue for guiding visual search [10] and conveying command meaning [24].

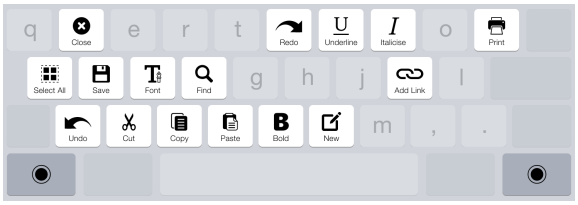

Fig. 17. Layout of our realistic command set, taken from the Windows version of Microsoft Word.

6.2 Task

This study relied on a simple command selection task. At the beginning of each trial, the name of the target command would be displayed on top of the screen (Figure 16a). Participants thus had to select the command by hitting a modifier ⌘ key to reveal the soft keyboard (Figure 16b) and then pressing the target command key. Upon selection, participants would receive both audio and visual feedback for 500ms to indicate success or failure (Figure 16c). Participants were instructed to perform the task as fast and accurately as possible. For each trial, time was measured as the time between stimulus appearance to when the correct target command was selected.

6.3 Stimuli and Command Mappings

We tested two different command *mappings* (ABSTRACT and REALISTIC) of 16 commands each. The abstract set comprises of 16 countries (Figure 16-b,c), while the realistic set comprises of 16 existing hotkeys from Microsoft Word for Windows (Figure 17). The spatial layout of the abstract command layout was a vertical symmetry of the realistic command one. This ensures a similar distribution (in terms of rows and columns) and a comparable average distance between the modifier keys and the command keys. We also avoided mnemonic associations between keys and countries, e.g. A for Australia. 8 commands from each set were chosen as targets: Australia, Brazil, Canada, Denmark, Germany, Iran, South Africa, and Vietnam for abstract set and Add Link, Cut, Find, Paste, Print, Save, Underline, and Undo for the realistic set. We decided to avoid having the same key used as a target in both conditions (e.g. X mapped to United States and Cut) to prevent users from potentially creating memorisation strategies by associating realistic commands with abstract commands.

6.4 Procedure

Participants began the experiment by signing a consent form and completing a questionnaire about their demographic information, handedness and usage of tablets and Microsoft Word. They were then evaluated on their knowledge of the 16 commands from the realistic set. For each mapping condition, learning is facilitated when participants performed 1 training block (B0) followed by 8 test blocks (B1-B8). Each block consists of 16 trials, where each of the 8 chosen target commands was repeated twice in random order. We ensured that participants did not accumulate fatigue by recommending ample breaks between blocks. Participants would also need to provide subjective ratings on each mapping after the completion of each condition. Finally, we measured retention rates on the 8 abstract and realistic commands at multiple *phases*: at the end of the study trials and after 1, 3 and 7 days. Each retention phase is facilitated through a form and requires less than 2 minutes. In the form, we displayed an ordinary keyboard layout, and the 16 target commands (with icons) used during trials (8 for each mapping), and we asked participants for the keys mapped to each command.

6.5 Participants and Apparatus

We recruited 12 participants (8 female, 4 male, all right-handed), aged 19 to 29 (M=24.9, SD=2.9) from the university community. All 12 participants were not involved in previous experiments. All of them had used Microsoft Word before, while 4 of them had never used a tablet. 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 Galaxy Tab 4 (10.5" screen, 483 grams) running Android 9 for the experiment, and each participant received $5 reimbursement for their participation.

We asked each participant if they knew the hotkey mapped to the 16 commands from our REALISTIC set. None of the 12 participants knew the shortcut mapped to Add Link command [k], 9 knew both Underline [u] and Undo [z], and all knew the rest of the commands. This means that out of the 8 target commands, all our participants had to learn only one of them (i.e. Add Link). 3 participants had difficulties assigning the key [u] between Underline and Undo commands since both share the same starting letter. Participants could rely on prior knowledge and minimise the slower visual search for the remaining 5 targets (Cut, Find, Paste, Print, and Save). On the other hand, participants had no prior knowledge of all 8 target commands for the ABSTRACT set and had to learn them through the trials.

6.6 Design

We used a within-subject design with MAPPING as a primary independent variable with 2 levels { ABSTRACT, REALISTIC }, and BLOCK and PHASE as secondary independent variables. BLOCK has 9 levels { B0-B8 } while PHASE has 4 levels { B0-B8 } while PHASE has 4 levels { 0 DAY (0D), 1 DAY (1D), 3 DAYS (3D), 7 DAYS (7D) }. We fully counterbalanced the order of MAPPING being presented to the participants. In terms of dependent variables, we measured the *time* and *accuracy* of each trial and the *retention rate* of each MAPPING. Note that *accuracy* measures the learning stage, while *retention rate* measures the recall stage. A trial was considered successful if a participant was able to select the correct command. We measured accuracy as the ratio between successful trials and the total number of trials for each block. We also measured subjective feedback using a 1-5 Likert Scale (1: strongly disagree, 5: strongly agree) on their overall opinion ("I liked using this mapping") between abstract and realistic command mapping, in terms of *ease of use*, *speed*, *accuracy* and how this mapping leveraged their *prior knowledge* of hotkeys. The participants completed both learning and recall stages in approximately 30 minutes over one week. There were no dropouts throughout the multiple sessions. In summary, we recorded 12 participants × 2 mappings × (1 training block + 8 test blocks) × 8 commands × 2 repetitions = 3456 trials in total.

6.7 Results

We first marked each trial as an outlier if its trial *time* was above or below 3 standard deviations away from the mean *time*. A total of 63 trials (1.8%) were removed for subsequent time analysis. 32 of which are from abstract mapping and 31 of which are from realistic mapping. Since our *time* measurements significantly deviated from normality, we applied log transformations. We also transformed *accuracy* data using Aligned Rank Transform [76]. We used a MAPPING × BLOCK ANOVA, followed by pairwise t-tests with Bonferroni corrections for post hoc comparisons. When the assumption of sphericity was violated, we corrected both p-values and degrees of freedom using Greenhouse-Geisser ($\epsilon < 0.75$). For every dependent variable, except subjective rating, trials were aggregated by participant and factors being analysed. For subjective rating, we used the Wilcoxon signed-rank test for analysis.

6.7.1 Learning Effect. There is a significant main effect of BLOCK ($F_{8,88} = 23.40$, $p < .0001$, $\eta^2_G = .19$) on *time*, but no interaction effects involving BLOCK. Pairwise comparisons found that training block B0 took significantly more time

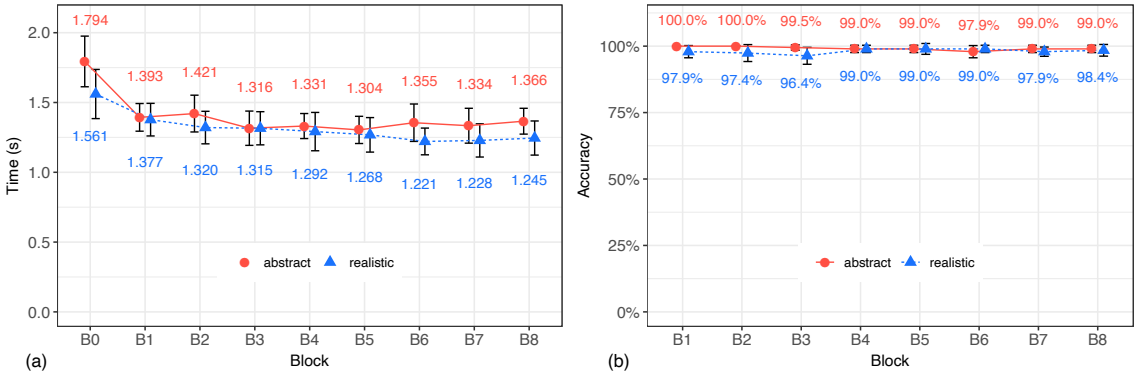

Fig. 18. (a) *Time* by BLOCK for each MAPPING. (b) *Accuracy* by BLOCK for each MAPPING. Error bars are 95% confidence.

to complete than all test blocks B1-B8 (all $p < .001$). Hence, we removed B0 for the main analysis but included a brief analysis of *time* for training data only below. In the subsequent analysis, we only used the 8 test blocks (B1-B8) since they are more representative of practised performance.

6.7.2 Time. For test blocks, we found that participants selected realistic commands faster than abstract commands. There were significant main effects of MAPPING ($F_{1,11} = 7.87$, $p = .017$, $\eta^2_G = .03$) and BLOCK ($F_{7,77} = 2.81$, $p = .012$, $\eta^2_G = .03$) on *time* but no interaction effect between MAPPING and BLOCK ($p > .05$). However, pairwise comparisons did not show any significant difference between the test blocks (all $p > .05$). The significant difference between ABSTRACT (M=1.352s) and REALISTIC (M=1.283s) commands was 69ms (about 5% of average trial time). As illustrated in Figure 18a, realistic commands require consistently less time on average than abstract commands across all blocks, but the difference remains marginal. For training block only, we found a non-significant ($p = .099$) 233ms difference between ABSTRACT (M=1.794s) and REALISTIC (M=1.561s) conditions. Therefore, this result **supports H3a (participants are faster when selecting commands from a mapping they are familiar with)**.

6.7.3 Accuracy. We found a significant main effect of MAPPING on *accuracy* ($F_{1,165} = 5.32$, $p = .022$, $d = 0.23$) but no interaction effect between factors ($p > .05$). Participants were highly accurate in both MAPPINGS, with accuracy for ABSTRACT commands (M=99.2%) higher than that of REALISTIC (M=98.1%) (Figure 18b). Therefore, this result **rejects H3b (participants are more accurate when selecting commands from a mapping they are familiar with)**.

6.7.4 Retention Rate. In terms of retention rate, participants could recall the spatial layout for REALISTIC commands (M=95.1%) more effectively than that of ABSTRACT commands (M=55.7%) (Figure 19a) on average. We found significant main effect of MAPPING ($F_{1,11} = 47.19$, $p < .01$, $\eta^2_G = .60$) and PHASE ($F_{3,33} = 3.01$, $p = .044$, $\eta^2_G = .06$) on *retention rate*. We also found an interaction effect between MAPPING and PHASE ($F_{3,33} = 4.83$, $p < .01$, $\eta^2_G = .09$). Simple main effect analysis revealed that retention for REALISTIC commands was higher than ABSTRACT commands for every PHASE condition (all $p < .001$) and retention for ABSTRACT commands was significantly higher during 0D than during 7D ($p = .047$). Therefore, this result **supports H3c (participants have a higher retention rate for mappings they are already familiar with as compared to newly learned mappings)**.

We further analysed the retention errors and noticed that majority of the errors were incurred when the participants remembered a command to be on a key adjacent to the actual target. For instance, participants recalled 'South Africa'

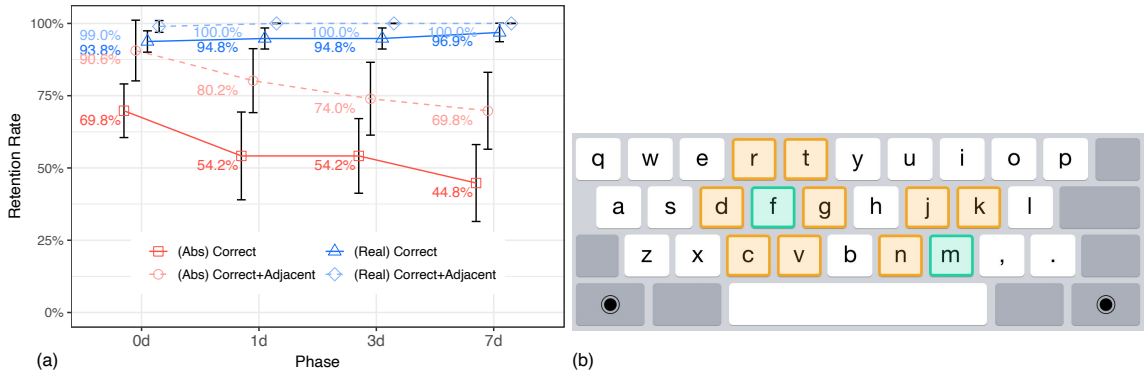

Fig. 19. (a) *Retention rate* by PHASE (at the end of the experiment (0D), after 1 day, 3 days, and 7 days) for each MAPPING. "Correct" rates indicate that the participant recalled the right key, while "Correct+Adjacent" also includes recalls on one of the adjacent keys. Error bars are 95% confidence. (b) Participants tended to misplace shortcuts from the target location (*e.g.* in green) to one of the keys adjacent to the target (*e.g.* in orange).

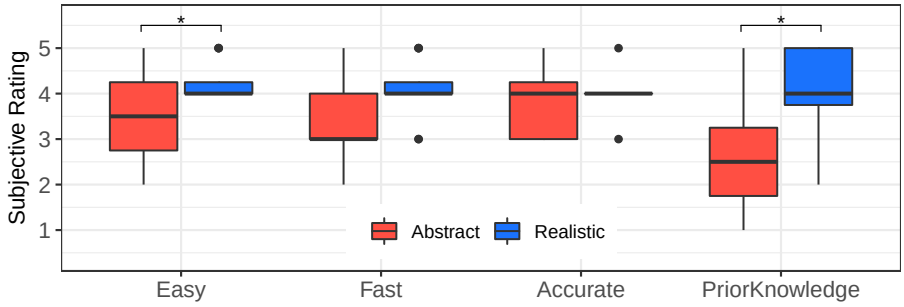

Fig. 20. Subjective rating for each MAPPING using a 5-point Likert scale (1: strongly disagree, 5: strongly agree).

M key on either the N, J or K keys (Figure 19b). Retention was also easier for targets (*e.g.* 'Brazil' or 'South Africa') with fewer adjacent keys. This suggests that while the retention for these targets is not perfect, participants could estimate the general area where the command was located even after one week.

6.7.5 Subjective Rating. Participants rated the REALISTIC command set as easier to use and leverage more prior knowledge than the ABSTRACT command set. There was a significant main effect of MAPPING ($V = 3.0$, $p = .036$) on *ease of use*, where 7 out of 12 participants rated REALISTIC (median=4.0) commands more favourably than ABSTRACT (median=3.5) commands. There was also a significant main effect of MAPPING ($V = 4.0$, $p = .015$) on leveraging *prior knowledge*, where 9 out of 12 participants rated REALISTIC (median=4.0) commands more favourably than ABSTRACT (median=2.5) commands.

6.8 Discussion

6.8.1 Small differences between Abstract & Realistic mappings. Despite the absence of prior knowledge, ABSTRACT mapping could still deliver high performance comparable to REALISTIC mapping. On average, during our test blocks,

participants were only 5% (69ms) slower in ABSTRACT mapping than in REALISTIC mapping. The time difference is likely due to a shorter visual search, as users were already familiar with the location of the key they were looking for. This is somewhat encouraging as it suggests that participants could be nearly as fast even with new mappings, suggesting that they switch mapping between applications effortlessly.

6.8.2 Similar learning effects across mapping. In addition, we observed a similar learning effect across the two mapping conditions. As we did not observe any interaction between BLOCK and MAPPINGS, we can conclude that the time performance gap we observed remained more or less consistent across time. However, more importantly, this lack of interaction also suggests that participants became almost equally fast with SoftCuts no matter which mapping was used, suggesting good potential for users to become expert with the technique rather quickly. Also, participants only required minimal training (one training block) to reach a low selection time. In the end, it is fair to assume that the performance difference measured during our study would fade out once users familiarise with the abstract mapping. We thus believe that SoftCuts could support expert performance regardless of the mapping used.

6.8.3 Role of spatial memory in long-term recall. It was rather expected for participants to remember REALISTIC command mappings more easily due to years of exposure and practice. However, we were surprised to learn that with only about 15 minutes spent by each participant to complete the 9 blocks of trials in our study, participants could estimate the general command location even after 1 week without rehearsal. Their estimations were spatially accurate, considering that their answers were directly adjacent to the target key being asked to recall. This result is of particular interest, as it means that users would first gaze in the area of the keyboard where this command is located. If we were to include these adjacent key estimation as correct answers, ABSTRACT mapping observed retention of close to 70% of the commands after 1 week. This number will only increase with more prolonged and more frequent interaction. Hence, we are confident that SoftCuts not only can be generalised with any mapping across different applications but also accelerate users' novice-to-expert transition.

6.9 Limitations

One of the challenges for memory recall is when the retrieval of a given piece of information suffer from interference by another similar piece [3]. Parnas [60] highlighted how these could manifest in user errors and overall selection performance. In the context of our hotkeys mapping, the inconsistencies across applications and devices may result in two possible scenarios. It is either 1) the same key mapped to multiple commands or 2) the same command mapped to different keys. For the first scenario, Scarr et al. [67] conducted a study that suggested participants have a strong knowledge of the interface despite having several commands overlapped at the same location. For the second scenario, this challenge applies to virtually any technique, not only SoftCuts. Therefore, future studies need to consider how we could standardise mappings across applications and devices as much as possible.

7 STUDY 4: COMPARISON BETWEEN GRID AND KEYBOARD LAYOUT

The previous studies have shown that SoftCuts have great potential to be used as a unified command selection mechanisms on touch-based devices. To this day, another technique has been shown to be fast and accurate on many touch-based devices: FastTap [31]. As we discussed in the motivation section, FastTap and SoftCuts are quite different. FastTap, a rehearsal-based interface (RBI), requires a delay, typically of 250ms, before displaying the available commands and does not leverage any prior knowledge of hotkeys. The main advantage of FastTap comes from the full-screen grid layout that takes advantage of large screens, whereas SoftCuts use a space limited to a small keyboard layout. As

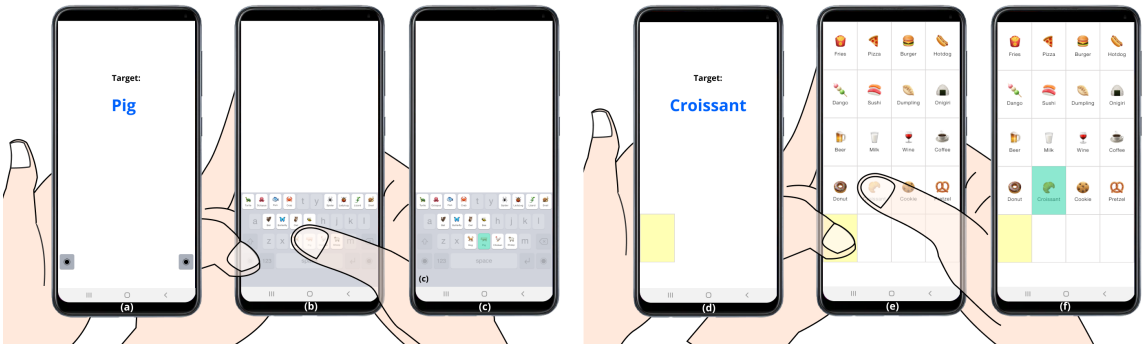

Fig. 21. Demonstration of how a selection trial is completed by each participant on the phone prototype for both spatial layouts: keyboard (a to c) and grid (d to f). (a) and (d) is when the target stimulus is first presented while the modifier key(s) is/are placed near the bottom of the screen, then (b) and (e) is when the participant uses one of their finger to hold onto a modifier key while using another finger to press the target command. (c) and (f) show the visual feedback when the selection is completed correctly.

such, FastTap offers individual areas for each command that are both larger and more spread on the screen. As the performance of a technique is important for adoption, we thus wondered how this layout difference would impact selection accuracy and speed for SoftCuts. Therefore, comparing a keyboard layout to a full-screen grid-based layout would help to quantify the magnitude of the impact that the choice of relying on a keyboard layout has on performance.

7.1 Layout Conditions

In order to offer a fair comparison, we implemented a full-screen grid layout similar to FastTap, but we changed a few points detailed below. First, to minimise potential confounding effects stemming from the activation delay of FastTap, we ensured that our grid layout is not at a disadvantage by removing it [26, 31]. This means that the system would immediately display the commands (grid or keyboard layout) upon pressing the modifier key. Both the grid and keyboard layouts adopted the user-maintained input method used in the original FastTap [31]. The commands only appear as long as the modifier key is pressed and disappear as soon as the modifier key is released. We did not force the participants to perform a selection using a chording gesture, and they were also allowed to hold the devices with two hands. The grid layout used relies on a 4×5 grid layout, with the first 4 rows used to display 16 commands as in the original FastTap. In contrast, SoftCuts only used the same amount of space as an on-screen keyboard. Since the commands on the grid layout are all next to each other, we created a mapping with four commands adjacent to each other (Figure 21) for the keyboard layout. This puts our keyboard layout at an additional disadvantage, as it may make selection harder on the comparatively smaller targets.

7.2 Task

For both grid and keyboard layouts, the target command name would be displayed in the middle of the screen (Figure 21) at the beginning of each trial. Participants then had to select the command via a user-maintained mode, that is, by first holding a modifier key with one finger to reveal the command layout and then pressing the target command key. Note that we decided to rely on the user-maintained activation method to remain in line with the FastTap design. As a reminder, previous work[21] suggests that it is a sub-optimal activation method for SoftCuts and possibly for FastTap (but it has not been investigated). However, contrary to FastTap [31], participants were not forced to perform only chord gestures when selecting a command and could use both hands for both layouts. Participants would receive both

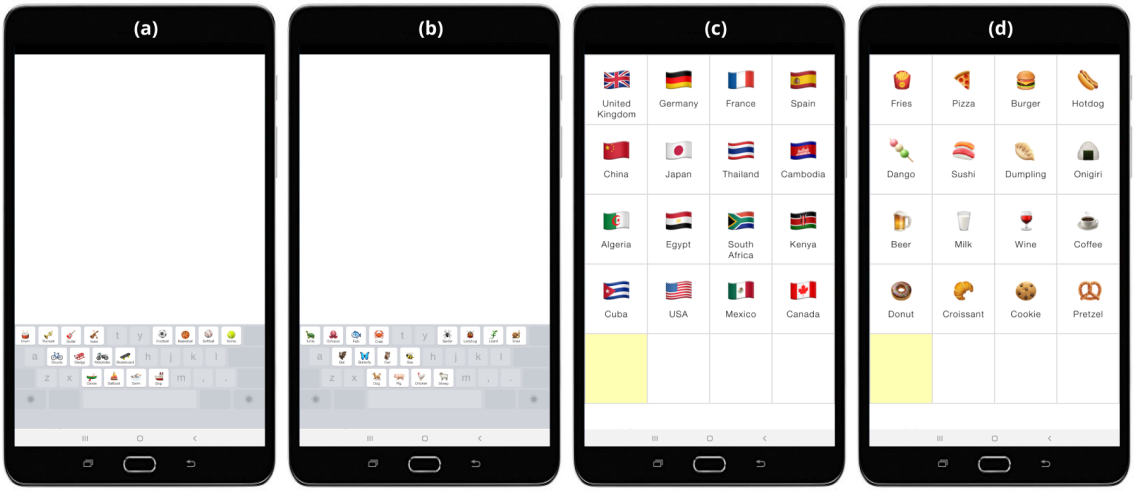

Fig. 22. Layouts displaying (a) activity, (b) animal, (c) flag, and (d) food command mappings on a tablet prototype. (a) and (b) use a keyboard layout while (c) and (d) use a grid layout (similar to FastTap).

audio and visual feedback for 500ms to indicate success or failure upon each selection. The green colour was used for a correct selection, while red was used for an incorrect selection. In order to guarantee a fair comparison, we hid the target name as soon as the modifier key has been pressed in both grid and keyboard conditions. We instructed each participant to perform the selection task as fast and accurately as possible. For each trial, time was measured from when the target command was first displayed to when the participant invoked a selection.

7.3 Stimulus

We used 4 different command mappings (activity, animal, flag, food) of 16 commands each (Figure 22). Activity and animal mappings are reserved for the keyboard layout, while flag and food mappings are reserved for the grid layout. All 4 mappings contain 4 sub-groups to categorise similar commands together in each row, *e.g.* for flag mapping, we have diverse countries from Europe, Asia, Africa and North America. This strategy was used in FastTap [31]. We similarly applied it to the keyboard layout by having 4 sets of 4 adjacent keys distributed across the 3 rows available on the keyboard layout. This was to ensure that the spatial arrangement of commands between both layouts would be similar and would not result in potential confounding effects on performance. For the keyboard layout, we avoided mnemonic associations between keys and activities/animals, *e.g.* "P" for Pig. 8 commands out of the total 16 were chosen as targets from each mapping set.

7.4 Procedure

Participants began the experiment by signing a consent form and completing a questionnaire about their demographic information, handedness, type of smartphone, and usage frequency of tablet. Then, each participant had to complete the selection tasks across 4 conditions combining device, layout and mapping variant. In terms of device, we used both a smartphone and a tablet in portrait orientation. For each device, participants had to complete the selection tasks with both layouts, one layout after the other. This can be done by counterbalancing the mapping associated with the same device and layout combination. For instance, P1 and P2 had the same order for Device and Layout while the

mapping was swapped accordingly. For each of the 4 conditions, participants performed 1 training block (B0) followed by 5 test blocks (B1-B5). We ensured that participants did not accumulate fatigue by recommending ample breaks between conditions. Each block consisted of 16 trials, where each of the 8 chosen target commands was repeated twice in random order. We also asked participants to subjectively rate their preferences between both layouts using a Likert Scale after each device condition had been completed. This is so that their perceptions would not suffer from biases across different devices.

7.5 Participants and Apparatus

We recruited 16 participants (8 female, 8 male and all right-handed), aged 19 to 30 (M=22.8, SD=3.4) from the university community. All 16 participants were not involved in previous experiments. 10 of them used an Android smartphone, while 6 of them used iOS smartphone. Only 2 of them had never used any tablet while 5 used daily, 1 used weekly, 4 used monthly and another 4 used yearly. 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 Galaxy Tab 4 (10.5" screen, 483 grams) running Android 9 for the tablet device condition of the experiment and a Samsung Galaxy A10 (6.2" screen, 168 grams) for phone device condition. Each participant received $5 as reimbursement for their participation.

7.6 Design

We used a within-subject design with LAYOUT and DEVICE as primary independent variables and BLOCK as a secondary independent variable. LAYOUT has 2 levels { GRID, KEYBOARD }, DEVICE has 2 levels { PHONE, TABLET } and BLOCK has 6 levels { B0-B5 }. The order for the device and layout were counterbalanced to minimise any potential order effect. In terms of dependent variables, we measured the *time* and *accuracy* of each trial. We measured subjective feedback using a 1-5 Likert Scale (1: strongly disagree, 5: strongly agree) on their overall opinion ("I liked using this technique") between both layouts, and also in terms of *ease of use*, *speed*, *accuracy*, *comfort*, and *memorability*. The experiment took approximately 30 minutes to complete. In summary, we recorded 16 participants × 2 techniques × 2 devices × (1 training block + 5 test blocks) × 8 commands × 2 repetitions = 6,144 trials in total.

7.7 Results

We first marked each trial as an outlier if its trial *time* was above or below 3 standard deviations from the mean *time* of its respective device and layout condition. A total of 97 trials (1.6%) were removed for subsequent analysis. 34 of which from KEYBOARD-PHONE condition, 15 of which from KEYBOARD-TABLET condition, 27 of which from GRID-PHONE condition, and 21 of which from GRID-TABLET condition. Since *time* measurement significantly deviated from normality, we applied log transformations. Similarly, we used Aligned Rank Transform [76] on *accuracy*. We used pairwise t-tests with Bonferroni corrections for post hoc comparisons. Trials were aggregated by participant and factors being analysed. When the assumption of sphericity was violated, we corrected both p-values and degrees of freedom using Greenhouse-Geisser ($\epsilon < 0.75$). For subjective rating, we used Wilcoxon signed-rank test for analysis. Additional analysis using ANOVA showed that the icons used in each LAYOUT do not contribute any difference (all $p > .05$) in performance between LAYOUTS.

7.7.1 Learning Effect. There was a significant main effect of BLOCK ($F_{5,75} = 101.6$, $p < .00001$, $\eta^2_G = .42$) on *time*. Pairwise comparisons revealed that participants require more time during training block B0 and test block B1 (all $p > .05$) as compared to the remaining blocks. There was also an interaction effect between LAYOUT, DEVICE, and BLOCK ($F_{5,75} = 2.45$,

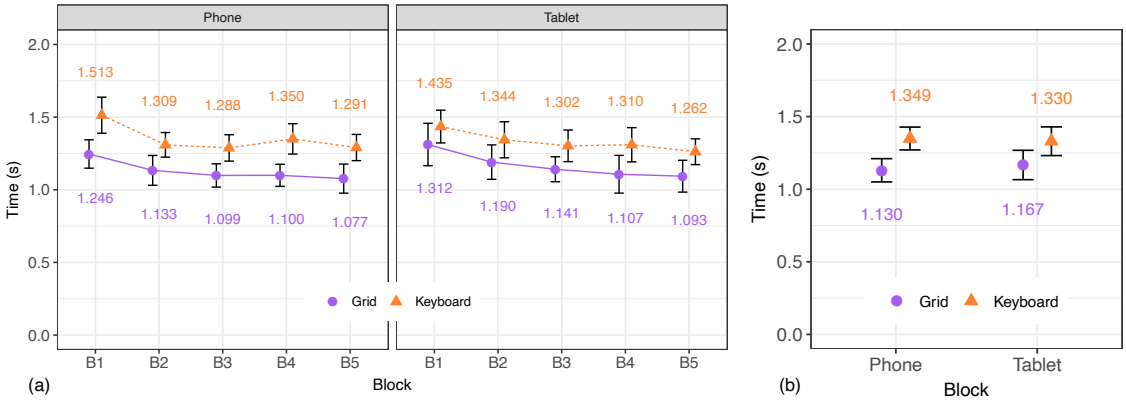

Fig. 23. (a) *Time* by BLOCK for each LAYOUT and DEVICE. (b) *Time* by DEVICE for each LAYOUT while excluding blocks that demonstrated learning effect. Error bars are 95% confidence.

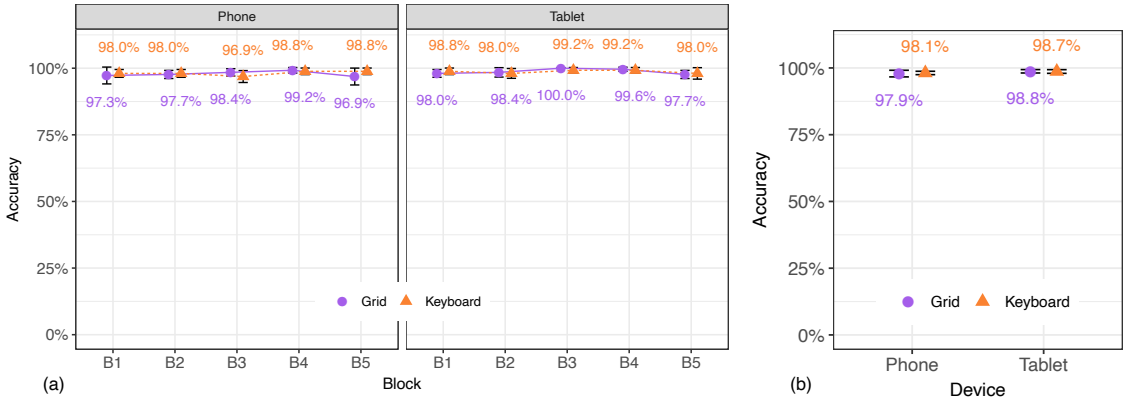

Fig. 24. (a) *Accuracy* by BLOCK for each LAYOUT and DEVICE. (b) *Accuracy* by DEVICE for each LAYOUT. Error bars are 95% confidence.

$p = .041$, $\eta^2_G = .01$). For PHONE condition, the KEYBOARD was significantly slower in test block B1 than in B3 ($p = .026$) and B5 ($p = .029$) while GRID showed no significant difference between B1 and all other blocks. For TABLET condition, both layouts showed that B1 required more time than all other blocks (all $p < .05$). These results interestingly suggest that TABLET condition may require more learning than PHONE condition. Therefore, we only removed training block B0 from our subsequent analysis.

7.7.2 Time. We found a significant main effect on time of LAYOUT $F_{1,15} = 38.8$, $p < .0001$, $\eta^2_G = .19$ with GRID requiring less time than the KEYBOARD for both PHONE (219ms or 16.2% faster) and TABLET (164ms or 12.3% faster) conditions (Figure 23b). We did not find any interaction between LAYOUT, DEVICE and BLOCK. **This result supports H4a (selection time is lower on a grid layout as compared to a keyboard layout on tablets and phones)**. We also modelled our interaction in each LAYOUT using [25], and its predictions match that with ours.

7.7.3 Accuracy. Participants were highly accurate with both LAYOUTS while using each DEVICE, with an average ranging from 97.9% to 98.8% (Figure 24b). We found no significant main effect of LAYOUT ($p = .79$) or DEVICE ($p = .54$) on *accuracy*

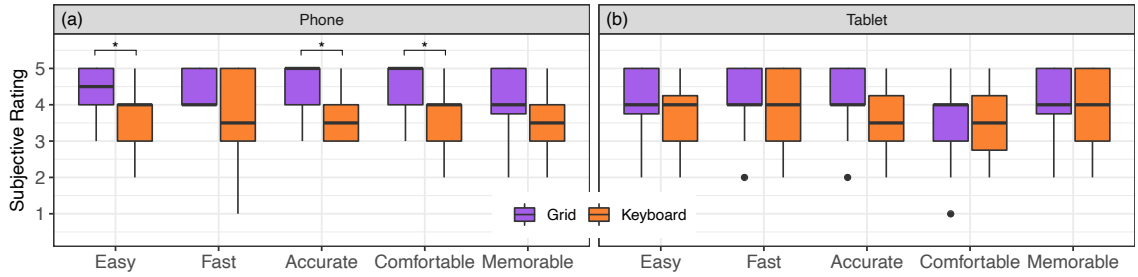

Fig. 25. Subjective rating for each LAYOUT grouped by DEVICE: (a) PHONE and (b) TABLET (1: Strongly Disagree, 5: Strongly Agree).

and no interaction. Interestingly, this allows us **to reject H4b (selection accuracy is higher on a grid layout as compared to a keyboard layout on tablets and phones)**. These results suggest that the accuracy of both layouts are similarly high despite the fact that the KEYBOARD layout occupies less than a third of the space occupied by the GRID.

7.7.4 Subjective Rating. Participants rated that using the GRID is easier, more accurate and more comfortable than using a KEYBOARD while on the PHONE but perceived both layouts as similar while on TABLET (Figure 25). For phone, there was a significant main effect of LAYOUT ($V = 83.5$, $p = .048$) on *ease of use*, where 14 out of 16 participants rated GRID (median=4.5) more favourably than the KEYBOARD (median=4.0). There was another significant main effect of LAYOUT ($V = 80.0$, $p = .014$) on *accuracy*, where 15 out of 16 participants rated GRID (median=5.0) more favourably than the KEYBOARD (median=3.5). There was another significant main effect of LAYOUT ($V = 79.0$, $p = .014$) on *comfort*, where 10 out of 16 participants rated GRID (median=5.0) more favourably than the KEYBOARD (median=4.0). For tablet, we did not find any significant main effect of LAYOUT on any measure. Overall, 11 out of 16 participants chose a GRID layout over a KEYBOARD as their preferred LAYOUT for each DEVICE.

7.8 Discussion

7.8.1 Limited speed benefit of a Grid layout. Our results revealed a clear performance gap between the speed of a grid layout (160-220 ms faster) and a keyboard. This was expected when one considers that this advantage comes at the cost of full-screen occlusion, making the pointing task of selecting a command easier for the grid layout. While Gutwin et al. [31] suggested using partial transparency to alleviate this problem, the visual conflict between actual content and command layout may not be easy to ignore, posing a serious challenge to eventual adoption. More recently, work from Henderson et al. suggested that visual occlusion might not be a problem with Marking Menus (another type of RBI). However, it is unclear whether these results would apply to full-screen interfaces like FastTap [32].

It is important to note that the grid layout we implemented did not use the advocated delay (of 250 ms) before displaying the grid layout, potentially reducing the performance gap between SoftCuts and the actual FastTap. Adding this delay would reduce and might even negate the time advantage of FastTap over SoftCuts. This delay would also be a long term problem for users: research conducted specifically on the adoption of expert mode with FastTap revealed difficulties to adopt this mode [26, 30, 40], suggesting that this delay will always impact a decent proportion of commands. In addition, the performance of the keyboard layout could be improved by using a faster input method such as Once [21], and spreading the shortcuts instead of using 4 groups of 4 adjacent commands. Hence, despite its slightly higher command selection time, a keyboard layout presents a desirable alternative, especially when each of its keys consumes

88-90% less screen real estate for a 12-16% speed cost as compared to the grid layout. Finally, it is worth noting that there was no benefit of the grid layout over a keyboard layout in terms of accuracy.

7.8.2 Smaller screen estate is an advantage for the keyboard layout. To our surprise, the keyboard layout did not suffer from any accuracy drop despite having a much smaller margin of error (reported by P2,6,9) due to the smaller keyboard layout. In fact, both layouts similarly demonstrated a high accuracy range between 97-99% for both phones and tablets. Although our subjective rating showed that participants rated the grid layout as more accurate and comfortable to use than the keyboard layout, we found conflicting remarks by some participants when we asked them to elaborate on their reasons. They shared that *"having the keys distributed over a larger surface area [with the grid layout] means that my fingers had to move over a bigger distance"* [P2,3,5,8]. This reachability issue was not observed when using the keyboard layout because participants *"could use minimal wrist movement and hence require less effort"* [P3,7,15] to reach all corners of the keyboard. The keyboard layout induced a bimanual posture of interaction that was found to be *"more comfortable"* [P1,2] and allowed P9, P14, and P16 to support the weight of the tablet by *"anchor[ing] [their] thumb"* [P9]. Therefore, the keyboard offers a clear advantage in terms of comfort as compared to grid layout for implementations across multiple platforms.

7.8.3 Keyboard layout supports greater spatial stability. While both layouts incorporate spatial stability in their designs, we have to acknowledge the challenge of integrating each layout across different applications and device sizes. The grid-based strategy does not have to follow a standardised mapping, unlike our keyboard-based strategy. For instance, a grid layout would require more effort to ensure that designers and developers from different applications choose a consistent grid size and position within the grid for the same command implemented. Also, Gutwin et al. [31] suggested that it is possible to "have a grid of up to 40 buttons in an 8x5 layout on a 7-inch tablet". This flexibility may result in greater spatial instability of commands across devices and reduced performance if the individual area for each command becomes smaller. On the other hand, SoftCuts use a keyboard layout that ensures a useful constraint of 26 keys maximum (assuming each character key represents a command) regardless of the device being used. Spatial stability is also better preserved in our current implementation of SoftCuts because it can adopt the mappings that have been mostly standardised for physical keyboards for decades.

7.9 Limitations

We acknowledge that the actual-world context of text-editing or sketching application may impose additional constraints than what was introduced in our study. Future studies should consider investigating how different screen space occupied by the interface may impact user preference and actual task performance.

8 FINAL DISCUSSION

The results of our four studies show that SoftCuts can reliably be used as a unified command selection mechanism across various applications on phones and tablets.

8.1 Robust multi-device performance with minimal training

The results we observed in Study 3 and 4 show that SoftCuts are highly accurate and fast on both phones and tablets. Although we observed a 12-16% time difference with a grid layout (similar to FastTap), this gap could potentially be lowered by using an input method more adapted for SoftCuts (*i.e.* Once instead of User Maintained) [21]. Even with such a gap, SoftCuts offer higher spatial stability and take a reduced amount of space on the screen instead of the

whole screen. We believe this trade-off to be acceptable for users, especially when dealing with multiple mobile devices simultaneously for productivity-related tasks. Furthermore, the learning effect observed in Study 3 and 4 was mainly between the training block and the first subsequent test block, which means that users require minimal training to maximise performance. Coupled with the high accuracy observed, we advocate for SoftCuts to be used as a unified command selection method on touch-based devices with keyboard-based applications.

8.2 Icons are beneficial for beginners

Participants in the first study preferred a visual layout showing an icon and the command name (D3) associated with the key. This is especially because beginners who are new to hotkeys mapping can rely on their prior knowledge of icons used in mainstream applications like Microsoft Word. Although this comes at the cost of not displaying the letter on the key (like in D1 and D2), using icons help to facilitate visual search [10] and convey command meaning [24]. In fact, our results from the first study show that D3 was rated more positively in many aspects as compared to D1, and at least 9 out of 12 participants chose D3 over D2 as the overall preferred design. Also, previous work on memorisation [62] suggests that combining spatial memory and images may help long term retention.

8.3 Letters may be more suitable for experts

Participants also noted that they might prefer the visual design (D2) with the letter and the command name for mappings that they are more familiar with. Besides, looking at the retention rate in Study 3, we noted that participants leveraged spatial memory to remember the general location of a command instead of the exact key location. This may come from the visual design (D3) we used in that study. Using D2 (letter + command name) could lead to better and more precise retention, though more work would be needed to confirm this hypothesis. It could thus be beneficial to allow users to customise their visual representation on the actual implementation of the technique, either for the entire system-wide (for consistency) or on an app basis (where users could use icons for apps they do not use often).

8.4 Use a familiar modifier key

Our study is the first one investigating discoverability for SoftCuts, and it is thus hard to compare to any other techniques. While our results show that familiarity and saliency of modifier key do not affect the discoverability rate, participants shared qualitative benefits of using ⌘ / Ctrl as compared to ⌥. Therefore, we recommend using a familiar modifier key that matches the OS: ⌘ for iPhones or iPads and Ctrl for Android or Windows devices.

8.5 Users can learn multiple mappings

Study 3 demonstrated that participants could remember two different mappings for a week. While they could leverage prior knowledge for the realistic mapping, the retention for the abstract mapping remained high, despite a very short exposure to the mapping. In real life, it would be expected that users would not spend an entire week without using or seeing SoftCuts on the apps they use frequently. In that study, the two mappings illustrate two extreme scenarios: retention for a mapping with limited training (abstract) vs. retention of a well-known mapping (realistic). If we thus consider abstract as a worst-case scenario, retention would increase to reach the performance of the realistic condition, as long as there is regular usage of SoftCuts.

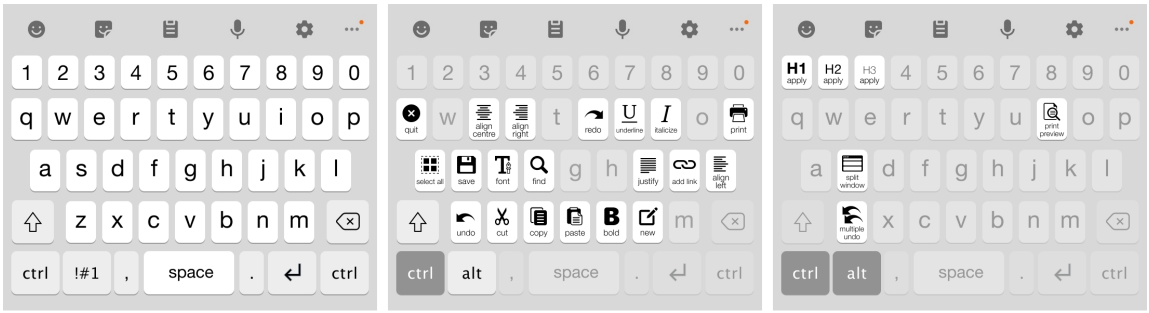

Fig. 26. (a) Example of SoftCuts on an Android soft keyboard with a Ctrl modifier key. (b) Tapping Ctrl once displays the first level of shortcuts (e.g. Ctrl + S for save), and substitutes !#1 with alt. (c) Tapping on alt displays additional shortcuts that are usually triggered with both Ctrl and alt modifier keys. For this illustration, the !#1 key is replaced by an alt key in command selection mode.

8.6 Users leverage prior knowledge of hotkeys

Our participants could take advantage of their prior knowledge gained from hotkeys on physical keyboards. While the advantage in terms of time performance was modest (5% time difference), it was still significant. The real gain was for retention, with a near-perfect retention rate after a week. Moreover, based on our Study 2 results, participants who were familiar with desktop hotkeys were more likely to discover the availability of SoftCuts spontaneously. This suggests that leveraging prior knowledge of hotkeys could impact not only performance of SoftCuts, but also their usage and adoption. As such, SoftCuts offer a clear advantage over any other touch-based command selection techniques.

8.7 SoftCuts for adoption of hotkeys on physical keyboards

While participants were able to leverage prior knowledge of hotkeys while using SoftCuts, we believe that a similar skill transfer could happen in the opposite direction. Currently, many younger users spend more time on a mobile device than on a desktop and may thus be exposed to SoftCuts before physical hotkeys. Using SoftCuts as a unified command selection mechanism on touch-based devices could thus teach users about hotkeys in general. It may allow them to leverage any knowledge of SoftCuts on desktop computers. SoftCuts users could potentially identify a familiar modifier key (⌘ or Ctrl) and then be tempted to press it on a desktop application. If that first step was to be completed, computers could use visual help to show possible shortcuts. Notably, we believe that techniques like the Chrome OS visual keyboard or KeyMap [45], which use a very similar visual representation by showing a keyboard on the bottom part of the screen of a desktop computer, could leverage users' familiarity with SoftCuts and help users get familiar with regular hotkeys. Finally, it is not unthinkable to envision the democratisation of physical keyboards that can dynamically update what their keys display, just like software keyboards.

9 FUTURE WORK

9.1 Expanding input vocabulary for SoftCuts

Our current work on SoftCuts only investigated the use of one modifier key (⌘). Hotkeys, in general, may use additional modifier keys, including Shift ⇧ and either Alt (Windows) or control and option (Mac OS). As Shift ⇧ is already present on every existing soft keyboard, it would be simple to add support for commands using it (e.g. ⌘ +

Shift ⇑ + Z for redo). Existing layouts could also be adapted to add more modifier keys (Figure 26). The changes to the layout would be minimal and could follow one of the two strategies:

1. Reducing the size of other keys (e.g. space) to add the additional modifier key(s) to the existing layout.
2. Replacing a key that is not used in command selection mode (i.e. when SoftCuts are active) and thus only affects the layout in that mode.

Using the first option would make it easier for users to discover SoftCuts, and suggest them to try using the keys. However, it could interfere with typing tasks and thus impact their performance. Replacing a key (e.g. from 123 to Alt) would keep the layout consistent, but it is unclear whether users would notice the subtle change. Another problem that would need to be investigated would be the feedforward mechanism to suggest additional shortcuts to be used on a given key. For example, how to show a user that typing on Z alone would undo, while typing on Shift ⇑ and Z would redo. Finally, supporting more modifier keys would also limit the use of the User Maintained input method, as maintaining three or more keys pressed at once may be hard if not impossible on touch-based devices.

9.2 Establishing mapping for new commands

With the increasing need to support large commands sets in feature-rich applications, it is equally important to identify which and how commands are to be mapped. For instance, Le et al. [43] recently elicited gestures for the 22 most important commands currently unavailable on smartphones. It would be interesting to conduct direct comparisons between the performance of keyboard-based (SoftCuts) and gesture-based shortcuts in selecting each of these commands. Future studies focusing on this aspect could reveal insights into how specific, less standardised commands could be facilitated more effectively by gestures instead of SoftCuts, potentially due to keybinding collisions.

10 CONCLUSION

In this work, we investigated SoftCuts as a unified command selection technique for phones and tablet. We first determined the best visual designs, and we found out that the optimal representation may vary depending on the users' level of expertise. We then investigated factors impacting discoverability in our second study and found out that while familiarity and saliency of modifier key do not affect our discoverability rates, using ⌘ / Ctrl helps those who are familiar with hotkeys to discover SoftCuts. Our third study showed that prior knowledge of a mapping between commands and keys might help in terms of selection time and retention and proved the benefits of spatial memory on retention for SoftCuts. Finally, we compared our keyboard layout with a grid layout and found out that despite a 12-16% performance drop in terms of time, the keyboard layout offers other advantages in terms of spatial stability and screen estate. Overall, our contributions reveal a promising future where command selections can and should be consistently facilitated across applications and devices for novices and experts alike. We also encourage designers and developers to leverage SoftCuts' versatility and integrate it into their respective applications.

11 ACKNOWLEDGEMENTS

This research/project is supported by the National Research Foundation, Singapore under its AI Singapore Programme (AISG Award No: AISG2-RP-2020-016). This work is also made possible 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 ACM Designing Interactive Systems Conference* (Eindhoven, Netherlands) *(DIS '20)*. Association for Computing Machinery, New York, NY, USA, 1103–1115. https://doi.org/10.1145/3357236.3395506

[2] John R. Anderson. 1983. *The Architecture of Cognition.* Harvard University Press, USA.

[3] John R Anderson and Gordon H Bower. 1974. Interference in memory for multiple contexts. *Memory & Cognition* 2, 3 (1974), 509–514.

[4] Caroline Appert, Olivier Chapuis, and Emmanuel Pietriga. 2012. Dwell-and-Spring: Undo for Direct Manipulation. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Austin, Texas, USA) *(CHI '12)*. Association for Computing Machinery, New York, NY, USA, 1957–1966. https://doi.org/10.1145/2207676.2208339

[5] Caroline Appert and Shumin Zhai. 2009. Using Strokes As Command Shortcuts: Cognitive Benefits and Toolkit Support. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Boston, MA, USA) *(CHI '09)*. ACM, New York, NY, USA, 2289–2298. https://doi.org/10.1145/1518701.1519052

[6] Apple. [n.d.]. https://developer.apple.com/design/human-interface-guidelines/ios/user-interaction/undo-and-redo/

[7] Jason Arndt. 2012. *Paired-Associate Learning.* Springer US, Boston, MA, 2551–2552. https://doi.org/10.1007/978-1-4419-1428-6_1038

[8] Jeff Avery and Edward Lank. 2016. Surveying Expert-Level Gesture Use and Adoption on Multi-Touch Tablets. In *Proceedings of the 2016 ACM Conference on Designing Interactive Systems* (Brisbane, QLD, Australia) *(DIS '16)*. ACM, New York, NY, USA, 577–581. https://doi.org/10.1145/2901790.2901895

[9] Gilles Bailly, Eric Lecolinet, and Laurence Nigay. 2007. Wave Menus: Improving the Novice Mode of Hierarchical Marking Menus. In *Proceedings of the 11th IFIP TC 13 International Conference on Human-Computer Interaction* (Rio de Janeiro, Brazil) *(INTERACT'07)*. Springer-Verlag, Berlin, Heidelberg, 475–488.

[10] 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

[11] Olivier Bau and Wendy E. Mackay. 2008. OctoPocus: A Dynamic Guide for Learning Gesture-based Command Sets. In *Proceedings of the 21st Annual ACM Symposium on User Interface Software and Technology* (Monterey, CA, USA) *(UIST '08)*. ACM, New York, NY, USA, 37–46. https://doi.org/10.1145/1449715.1449724

[12] 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

[13] Florian Block, Hans Gellersen, and Nicolas Villar. 2010. Touch-Display Keyboards: Transforming Keyboards into Interactive Surfaces. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Atlanta, Georgia, USA) *(CHI '10)*. Association for Computing Machinery, New York, NY, USA, 1145–1154. https://doi.org/10.1145/1753326.1753498

[14] 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

[15] John M. Carroll and Mary Beth Rosson. 1987. *Paradox of the Active User.* MIT Press, Cambridge, MA, USA, 80–111.

[16] Andy Cockburn, Carl Gutwin, Joey Scarr, and Sylvain Malacria. 2014. Supporting Novice to Expert Transitions in User Interfaces. *ACM Comput. Surv.* 47, 2, Article 31 (Nov. 2014), 36 pages. https://doi.org/10.1145/2659796

[17] Andy Cockburn, Per Ola Kristensson, Jason Alexander, and Shumin Zhai. 2007. Hard Lessons: Effort-Inducing Interfaces Benefit Spatial Learning. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (San Jose, California, USA) *(CHI '07)*. Association for Computing Machinery, New York, NY, USA, 1571–1580. https://doi.org/10.1145/1240624.1240863

[18] Charles Denis and Laurent Karsenty. 2003. *Inter-Usability of Multi-Device Systems – A Conceptual Framework.* John Wiley & Sons, Ltd, Chapter 17, 373–385. https://doi.org/10.1002/0470091703.ch17 arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1002/0470091703.ch17

[19] Alan Dix, Janet Finlay, Gregory Abowd, and Russell Beale. 1997. *Human-Computer Interaction.* Prentice-Hall, Inc., USA.

[20] Katherine Fennedy and Hyowon Lee. 2019. Multitouch Keyboard Revisited: Enhancing Moded Interaction Through Redesigning Structure and Switching Techniques. In *Proceedings of the 5th International ACM In-Cooperation HCI and UX Conference* (Jakarta, Surabaya, Bali, Indonesia) *(CHIuXiD'19)*. Association for Computing Machinery, New York, NY, USA, 36–45. https://doi.org/10.1145/3328243.3328249

[21] Katherine Fennedy, Sylvain Malacria, Hyowon Lee, and Simon Tangi 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* (Oldenburg, Germany) *(MobileHCI '20)*. Association for Computing Machinery, New York, NY, USA, Article 29, 12 pages. https://doi.org/10.1145/3379503.3403552

[22] Figma. [n.d.]. The Collaborative Interface Design Tool. https://www.figma.com/.

[23] Jeremie Francone, Gilles Bailly, Laurence Nigay, and Eric Lecolinet. 2009. Wavelet Menus: A Stacking Metaphor for Adapting Marking Menus to Mobile Devices. In *Proceedings of the 11th International Conference on Human-Computer Interaction with Mobile Devices and Services* (Bonn, Germany) *(MobileHCI '09)*. ACM, New York, NY, USA, Article 49, 4 pages. https://doi.org/10.1145/1613858.1613919

[24] Emmanouil Giannisakis, Gilles Bailly, Sylvain Malacria, and Fanny Chevalier. 2017. IconHK: Using Toolbar Button Icons to Communicate Keyboard Shortcuts. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems* (Denver, Colorado, USA) *(CHI '17)*. ACM, New York, NY, USA, 4715–4726. https://doi.org/10.1145/3025453.3025595

[25] Alix Goguey, Géry Casiez, Andy Cockburn, and Carl Gutwin. 2018. Storyboard-Based Empirical Modeling of Touch Interface Performance. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems* (Montreal QC, Canada) *(CHI '18)*. Association for Computing Machinery, New York, NY, USA, 1–12. https://doi.org/10.1145/3173574.3174019

[26] Alix Goguey, Sylvain Malacria, Andy Cockburn, and Carl Gutwin. 2019. Reducing Error Aversion to Support Novice-to-Expert Transitions with FastTap. In *Proceedings of the 31st Conference on l'Interaction Homme-Machine* (Grenoble, France) *(IHM '19)*. Association for Computing Machinery, New York, NY, USA, Article 1, 10 pages. https://doi.org/10.1145/3366550.3372247

[27] Alix Goguey, Sylvain Malacria, and Carl Gutwin. 2018. *Improving Discoverability and Expert Performance in Force-Sensitive Text Selection for Touch Devices with Mode Gauges.* Association for Computing Machinery, New York, NY, USA, 1–12. https://doi.org/10.1145/3173574.3174051

[28] Nathan Green, Jan Kruger, Chirag Faldu, and Robert St. Amant. 2004. A Reduced QWERTY Keyboard for Mobile Text Entry. In *CHI '04 Extended Abstracts on Human Factors in Computing Systems* (Vienna, Austria) *(CHI EA '04)*. Association for Computing Machinery, New York, NY, USA, 1429–1432. https://doi.org/10.1145/985921.986082

[29] Tovi Grossman, Pierre Dragicevic, and Ravin Balakrishnan. 2007. Strategies for Accelerating On-line Learning of Hotkeys. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (San Jose, California, USA) *(CHI '07)*. ACM, New York, NY, USA, 1591–1600. https://doi.org/10.1145/1240624.1240865

[30] Carl Gutwin, Andy Cockburn, and Benjamin Lafreniere. 2015. Testing the rehearsal hypothesis with two FastTap interfaces. In *Proceedings of the 41st Graphics Interface Conference*. 223–231.

[31] Carl Gutwin, Andy Cockburn, Joey Scarr, Sylvain Malacria, and Scott C. Olson. 2014. Faster Command Selection on Tablets with FastTap. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Toronto, Ontario, Canada) *(CHI '14)*. Association for Computing Machinery, New York, NY, USA, 2617–2626. https://doi.org/10.1145/2556288.2557136

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

[33] Alita Joyce. [n.d.]. Mobile-App Onboarding: An Analysis of Components and Techniques. https://www.nngroup.com/articles/mobile-app-onboarding/.

[34] Alita Joyce. [n.d.]. Mobile Tutorials: Wasted Effort or Efficiency Boost?s. https://www.nngroup.com/articles/mobile-tutorials/.

[35] Daniel Kahneman. 1973. *Attention and effort*. Vol. 1063. Citeseer.

[36] Jacob Kastrenakes. 2018. Swype keyboard has been discontinued. https://www.theverge.com/2018/2/21/17030216/nuance-discontinues-swype-keyboard.

[37] 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

[38] Brian Krisler and Richard Alterman. 2008. Training towards Mastery: Overcoming the Active User Paradox. In *Proceedings of the 5th Nordic Conference on Human-Computer Interaction: Building Bridges* (Lund, Sweden) *(NordiCHI '08)*. Association for Computing Machinery, New York, NY, USA, 239–248. https://doi.org/10.1145/1463160.1463186

[39] Gordon Kurtenbach and William Buxton. 1994. User Learning and Performance with Marking Menus. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Boston, Massachusetts, USA) *(CHI '94)*. Association for Computing Machinery, New York, NY, USA, 258–264. https://doi.org/10.1145/191666.191759

[40] 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

[41] Benjamin Lafreniere, Carl Gutwin, Andy Cockburn, and Tovi Grossman. 2016. Faster Command Selection on Touchscreen Watches. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems* (San Jose, California, USA) *(CHI '16)*. Association for Computing Machinery, New York, NY, USA, 4663–4674. https://doi.org/10.1145/2858036.2858166

[42] 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

[43] Huy Viet Le, Sven Mayer, Maximilian Weiß, Jonas Vogelsang, Henrike Weingärtner, and Niels Henze. 2020. Shortcut Gestures for Mobile Text Editing on Fully Touch Sensitive Smartphones. *ACM Trans. Comput.-Hum. Interact.* 27, 5, Article 33 (Aug. 2020), 38 pages. https://doi.org/10.1145/3396233

[44] G. Julian Lepinski, Tovi Grossman, and George Fitzmaurice. 2010. The Design and Evaluation of Multitouch Marking Menus. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Atlanta, Georgia, USA) *(CHI '10)*. ACM, New York, NY, USA, 2233–2242. https://doi.org/10.1145/1753326.1753663

[45] Blaine Lewis, Greg d'Eon, Andy Cockburn, and Daniel Vogel. 2020. KeyMap: Improving Keyboard Shortcut Vocabulary Using Norman's Mapping. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems* (Honolulu, HI, USA) *(CHI '20)*. Association for Computing Machinery, New York, NY, USA, 1–10. https://doi.org/10.1145/3313831.3376483

[46] Blaine Lewis and Daniel Vogel. 2020. Longer Delays in Rehearsal-Based Interfaces Increase Expert Use. *ACM Trans. Comput.-Hum. Interact.* 27, 6, Article 45 (Nov. 2020), 41 pages. https://doi.org/10.1145/3418196

[47] Sylvain Malacria, Gilles Bailly, Joel Harrison, Andy Cockburn, and Carl Gutwin. 2013. Promoting Hotkey Use Through Rehearsal with ExposeHK. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Paris, France) *(CHI '13)*. ACM, New York, NY, USA, 573–582. https://doi.org/10.1145/2470654.2470735

[48] Sylvain Malacria, Joey Scarr, Andy Cockburn, Carl Gutwin, and Tovi Grossman. 2013. Skillometers: Reflective Widgets That Motivate and Help Users to Improve Performance. In *Proceedings of the 26th Annual ACM Symposium on User Interface Software and Technology* (St. Andrews, Scotland, United Kingdom) *(UIST '13)*. ACM, New York, NY, USA, 321–330. https://doi.org/10.1145/2501988.2501996

[49] Sven Mayer, Lars Lischke, Adrian Lanksweirt, Huy Viet Le, and Niels Henze. 2018. How to Communicate New Input Techniques. In *Proceedings of the 10th Nordic Conference on Human-Computer Interaction* (Oslo, Norway) *(NordiCHI '18)*. Association for Computing Machinery, New York, NY, USA, 460–472. https://doi.org/10.1145/3240167.3240176

[50] Maze. [n.d.]. Design that works. https://maze.design/.

[51] Craig S. Miller, Svetlin Denkov, and Richard C. Omanson. 2011. Categorization Costs for Hierarchical Keyboard Commands. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Vancouver, BC, Canada) *(CHI '11)*. ACM, New York, NY, USA, 2765–2768. https://doi.org/10.1145/1978942.1979351

[52] Jakob Nielsen. 1993. *Usability Engineering*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.

[53] Donald A. Norman. 1983. Design Principles for Human-Computer Interfaces. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Boston, Massachusetts, USA) *(CHI '83)*. Association for Computing Machinery, New York, NY, USA, 1–10. https://doi.org/10.1145/800045.801571

[54] Donald A Norman. 1988. *The psychology of everyday things*. Basic books.

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

[56] 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

[57] 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-.

[58] 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. In *Proceedings of Graphics Interface 2004* (London, Ontario, Canada) *(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

[59] PandaSage1221. 2013. *Typing on Windows 8 / Windows RT Onscreen Touch Keyboard + Features*. Youtube. https://youtu.be/Dep_O1508Ug

[60] David L. Parnas. 1969. On the Use of Transition Diagrams in the Design of a User Interface for an Interactive Computer System. In *Proceedings of the 1969 24th National Conference (ACM '69)*. Association for Computing Machinery, New York, NY, USA, 379–385. https://doi.org/10.1145/800195.805945

[61] Kara Pernice and Raluca Budiu. [n.d.]. Hamburger Menus and Hidden Navigation Hurt UX Metrics. https://www.nngroup.com/articles/hamburger-menus/.

[62] Simon T. Perrault, Eric Lecolinet, Yoann Pascal Bourse, Shengdong Zhao, and Yves Guiard. 2015. Physical Loci: Leveraging Spatial, Object and Semantic Memory for Command Selection. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems* (Seoul, Republic of Korea) *(CHI '15)*. Association for Computing Machinery, New York, NY, USA, 299–308. https://doi.org/10.1145/2702123.2702126

[63] Nicole Ke Cheng Pong. 2020. *Interacting with signifier-less designs - the case of swhidgets*. Theses. Université de Lille (2018-....). https://hal.inria.fr/tel-03133122

[64] Nicole Ke Chen 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* (Daejeon, Republic of Korea) *(ISS '19)*. Association for Computing Machinery, New York, NY, USA, 193–204. https://doi.org/10.1145/3343055.3359713

[65] Lisa Posch, Arnim Bleier, Clemens M. Lechner, Daniel Danner, Fabian Flöck, and Markus Strohmaier. 2019. Measuring Motivations of Crowdworkers: The Multidimensional Crowdworker Motivation Scale. *Trans. Soc. Comput.* 2, 2, Article 8 (Sept. 2019), 34 pages. https://doi.org/10.1145/3335081

[66] 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/ie/support/mobile-devices/can-i-use-ctrl-c-ctrl-v-and-other-shortcuts-on-galaxy-tab-s-like-i-would-on-a-pc/.

[67] Joey Scarr, Andy Cockburn, Carl Gutwin, and Andrea Bunt. 2012. Improving Command Selection with CommandMaps. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Austin, Texas, USA) *(CHI '12)*. ACM, New York, NY, USA, 257–266. https://doi.org/10.1145/2207676.2207713

[68] Joey Scarr, Andy Cockburn, Carl Gutwin, and Sylvain Malacria. 2013. Testing the Robustness and Performance of Spatially Consistent Interfaces. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Paris, France) *(CHI '13)*. Association for Computing Machinery, New York, NY, USA, 3139–3148. https://doi.org/10.1145/2470654.2466430

[69] Joey Scarr, Andy Cockburn, Carl Gutwin, and Philip Quinn. 2011. Dips and Ceilings: Understanding and Supporting Transitions to Expertise in User Interfaces. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Vancouver, BC, Canada) *(CHI '11)*. ACM, New York, NY, USA, 2741–2750. https://doi.org/10.1145/1978942.1979348

[70] Katherine Schramm, Carl Gutwin, and Andy Cockburn. 2016. Supporting Transitions to Expertise in Hidden Toolbars. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems* (San Jose, California, USA) *(CHI '16)*. ACM, New York, NY, USA, 4687–4698. https://doi.org/10.1145/2858036.2858412

[71] Ben Shneiderman. 1992. *Designing the User Interface (2nd Ed.): Strategies for Effective Human-Computer Interaction*. Addison-Wesley Longman Publishing Co., Inc., USA.

[72] Herbert A Simon. 1956. Rational choice and the structure of the environment. *Psychological review* 63, 2 (1956), 129.

[73] Mark K. Singley and John R. Anderson. 1989. *The Transfer of Cognitive Skill.* Harvard University Press, USA.

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

[75] Daniel Vogel and Patrick Baudisch. 2007. Shift: A Technique for Operating Pen-Based Interfaces Using Touch. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (San Jose, California, USA) *(CHI '07)*. Association for Computing Machinery, New York, NY, USA, 657–666. https://doi.org/10.1145/1240624.1240727

[76] Jacob O. Wobbrock, Leah Findlater, Darren Gergle, and James J. Higgins. 2011. The Aligned Rank Transform for Nonparametric Factorial Analyses Using Only Anova Procedures *(CHI '11)*. Association for Computing Machinery, New York, NY, USA, 143–146. https://doi.org/10.1145/1978942.1978963

[77] Shengdong Zhao and Ravin Balakrishnan. 2004. Simple vs. Compound Mark Hierarchical Marking Menus. In *Proceedings of the 17th Annual ACM Symposium on User Interface Software and Technology* (Santa Fe, NM, USA) *(UIST '04)*. Association for Computing Machinery, New York, NY, USA, 33–42. https://doi.org/10.1145/1029632.1029639

[78] 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. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems* (Montreal QC, Canada) *(CHI '18)*. ACM, New York, NY, USA, Article 249, 14 pages. https://doi.org/10.1145/3173574.3173823