FingerArc and FingerChord: Supporting Novice to Expert Transitions with Guided Finger-Aware Shortcuts

Jingjie Zheng[1,2], **Blaine Lewis**[1], **Jeff Avery**[1], **Daniel Vogel**[1]
[1]Cheriton School of Computer Science, University of Waterloo, [2]Google
jingjie@acm.org, {blaine.lewis, j2avery, dvogel}@uwaterloo.ca

ABSTRACT

Keyboard shortcuts can be more efficient than graphical input, but they are underused by most users. To alleviate this, we present "Guided Finger-Aware Shortcuts" to reduce the gulf between graphical input and shortcut activation. The interaction technique works by recognising when a special hand posture is used to press a key, then allowing secondary finger movements to select among related shortcuts if desired. Novice users can learn the mappings through dynamic visual guidance revealed by holding a key down, but experts can trigger shortcuts directly without pausing. Two variations are described: FingerArc uses the angle of the thumb, and FingerChord uses a second key press. The techniques are motivated by an interview study identifying factors hindering the learning, use, and exploration of keyboard shortcuts. A controlled comparison with conventional keyboard shortcuts shows the techniques encourage overall shortcut usage, make interaction faster, less error-prone, and provide advantages over simply adding visual guidance to standard shortcuts.

Author Keywords
keyboard shortcuts; finger identification

INTRODUCTION

Design principles suggest that desktop interfaces should support keyboard shortcuts (also called "hotkeys") as an alternative method to issue commands [53, 62, 13]. Compared to point-and-click interaction, keyboard shortcuts are very fast due to the small motor action [61], and they enable parallel interaction, since keyboard and mouse input are combined [52, 57]. In the extreme, mastering a large set of shortcuts can even eliminate entire menus, de-clutter the interface, and increase screen space for the primary document of interest.

Given the advantages of shortcuts, it may be surprising that most users do not fully adopt them [28, 34, 47, 74]. To some extent, this can be ascribed to their poor visibility [28, 74], unintuitive command-to-key mappings [28, 34, 41, 61], sometimes cumbersome hand movements when using modifier keys [41, 52, 74], and lack of user motivation to learn efficient strategies [7, 8, 12, 54, 74]. More critically, the act of using shortcut keys is radically different than graphical input. This gulf hinders the effective transition to expert behaviour [34, 40, 41, 42, 43, 44].

We present *Guided Finger-Aware Shortcuts* to reduce the gulf between graphical input and shortcut activation. Finger-Aware Shortcuts [87] is a keyboard input method that maps multiple commands to the same key by using computer vision to recognise which finger, hand, and hand posture was used to press the key. We significantly extend that idea to using special postures for activating one of several command shortcuts using secondary finger movements with an optional trigger for dynamic visual guidance. Two variations of Guided Finger-Aware Shortcuts are presented and evaluated: FingerArc detects the angle of the thumb to select different commands (Fig. 1a); FingerChord detects a second key press on keyboard rows to select different commands (Fig. 1b).

Our work contributes a deeper understanding of current issues with keyboard shortcuts through an interview study with experts, the description and implementation of an expanded Finger-Aware Shortcut interaction space motivated by that understanding, and a comprehensive evaluation of the new techniques showing they encourage overall shortcut usage,

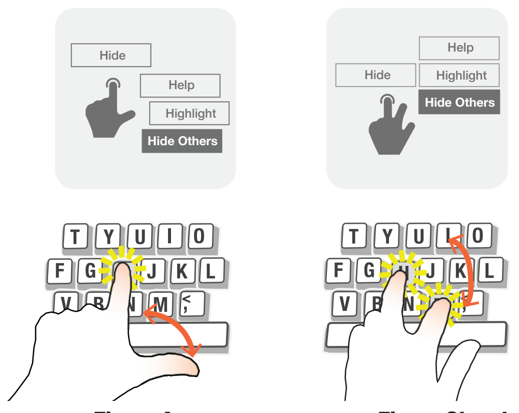

Figure 1: Guided Finger-Aware Shortcuts detect when a special hand posture is used to press a key, for example: (a) pressing H using the index finger with the little, ring, and middle fingers tucked enables FingerArc shortcuts, and holding the key down for a moment reveals guided feedback revealing the command options based on the angle of the thumb; (b) an alternative technique is FingerChord, enabled by pressing a key with the middle finger while little and ring fingers are tucked in. A long press reveals guidance explaining what commands can be invoked by pressing different keyboard rows with a second finger.

UIST'18, October 14–17, 2018, Berlin, Germany
© 2018 Copyright is held by the owner/author(s). Publication rights licensed to ACM. ACM ISBN 978-1-4503-5948-1/18/10...$15.00
DOI: https://doi.org/10.1145/3242587.3242589

enable faster and less error-prone command invocation, and may provide advantages over simply adding visual guidance to traditional shortcuts.

RELATED WORK

Our work is related to augmenting conventional input with expressive hand gestures, supporting novice to expert transitions, and fostering shortcut key learning.

Augmenting Keyboard Input with Expressive Gestures

Several works have pursued more expressive hand and finger input on various devices. At the finest level, a system may examine minute aspects of a single finger for specifying different actions such as orientation [24, 78, 79, 32], touch position [36], contact size [9], pressure [16], and proximity to the input surface [64]. Identifying *which* finger is in contact can also multiplex a single event when, for example, tapping on a touchscreen [15, 29, 23, 26, 25], a touchpad [51], pressing a button [80, 72], a piano key [27, 58], or a keyboard key [51, 87, 11]. Further, combining input from multiple fingers offers even more expressive power – users may not only control the number of fingers in contact [4, 21, 59], but also their individual positions [31] and trajectories [59, 3, 25]. At a coarser level, the system may detect subtle hand gestures that indicate an alternative input mode such as bringing together the thumb and the index finger [73], using the knuckle to knock [33], or holistic hand gestures such as rotation [11] and movement [85].

Our work encodes finger, hand, and hand posture recognition results as part of keyboard input so that the same key press can be interpreted in multiple ways. This is closely related to Finger-Aware Shortcuts that identify the finger and hand posture used when pressing a key [87] and Buschek et al.'s technique that combines keyboard shortcuts with rotation gestures [11]. The common goal of these projects is to increase expressivity for conventional input devices, although the way in achieving this varies. This paper demonstrates the feasibility of increasing shortcut expressivity using an initial "finger-aware" key press that differentiates among FingerArc, FingerChord, and regular text entry, plus optional secondary finger movements for specifying commands.

Supporting Novice to Expert Transitions

Another goal of our research is to make it easy for users to transition from slow point-and-click input to fast shortcut activation. This relates to the field of study that aims at supporting *novice to expert transitions* [13]. It is widely accepted that user interfaces should be easy to use for novices, include high-performance mechanisms for experts, and facilitate the progression from ease to efficiency [53, 62, 13, 86, 83]. However, abundant evidence shows that high-performance mechanisms are rarely fully leveraged [7, 8, 65, 55].

Carroll and Rosson [12] suggest this underuse may be inherent to our cognition. When faced with a high-level goal like writing an essay, spending time optimising individual steps like command activation is not the main interest. In addition, users tend to reuse methods like linear menus than learning shortcuts because they already know how to use them. In another words, there is a cost or "performance dip" [69] associated with learning more efficient methods that deters users from transitioning to experts. Our research aims to reduce this "dip" by making keyboard shortcuts easier to learn, and previous research has provided with many insights:

Rehearsal — Kurtenbach et al. [42] argue that in lieu of having completely different modalities for novice and expert interactions, the novice method should provide physical rehearsal for the expert behaviour. By using a single interaction technique, the need for a conscious and deliberate switch to a different modality is reduced to enable a seamless transition. In this spirit, Marking Menus [40, 42, 44, 43, 86] use the same directional stroke interaction for both novice and expert users. ShapeWriter [82, 39, 84, 83] transitions users from entering a word by tapping on individual letters to tracing through the letters with a single stroke gesture in a similar trajectory. Our techniques do not address the modality gap between menu and shortcut interfaces. Rather, we provide physical rehearsal for shortcuts by adding an intermediate mode for practising shortcut postures.

Guidance — Research suggests that providing an *appropriate* amount of guidance helps execution and learning [6, 42, 50, 21], whereas too much guidance may hinder them [70, 1]. For command activation, guidance means two things: using *feedforward* to show what actions can be specified, and using *feedback* to inform what action is currently active. For example, OctoPocus [6] displays contextual dynamic guides for possible pen-stroke gestures given partial strokes, while also showing which command will be triggered. Arpège [21] does the same, but for multi-finger chord gestures. Our techniques provide similar guidance based on the current key pressed and the active posture detected.

Effort — Evidence shows that an *appropriate* amount of effort added to novice interactions, such as a delay in revealing the guidance, motivates the use of expert methods [14, 17, 28, 86], whereas too much effort may harm usability [49, 30] and impede learning [14]. Our techniques impose a small delay before revealing visual guidance to subtly nudge users towards direct activation.

Risk Mitigation — Lafreniere et al. [45] suggest that the risk of making errors may hinder some users from transitioning to expert methods, even though the errors may be non-destructive. Peres et al. [61] also include risk as a contributing factor. We are not aware of any direct study examining this, but evidence seems to suggest providing mechanisms for mitigating errors may help expert transitions. Our techniques provide a cancellation gesture to mitigate risk, and we examine whether cancellation is leveraged.

Consistent Articulation — Abundant evidence shows that consistency in physical articulation leverages spatial memory to facilitate learning [66, 67, 68, 71, 18, 35]. For example, HandMark Menus [77, 75, 76] use the hand as landmarks to help memorise buttons around it. FastTap [30, 31, 46, 45, 20] guides novices to form a chord on a spatially stable grid of buttons. Our FingerChord technique has a similar interaction to FastTap, but in the context of physical keyboards.

Both our techniques provide consistent articulation since the same postures can be used to modify shortcut keys regardless of where the keys are. In contrast, conventional shortcuts require reaching to modifier keys at different distances.

It is worth noting that none of these strategies guarantee the transition to, and persistence of, expert techniques—they are all ways to facilitate it. Kurtenbach and Buxton [40] observe in a long-term study that users switch back and forth between novice and expert modes, and Lafreniere et al. [45] suggest post-training persistence of expert behaviour may require a global change in strategy and is heavily influenced by the contextual performance requirement.

Fostering Keyboard Shortcut Learning

The above strategies may all be applied to address the underuse of keyboard shortcuts [28, 34, 47, 74]. For example, to enable physical rehearsal, visual cues like a cheat sheet in Keycue [74] or shortcut information on an expanded menu in ExposeHK [49] can be revealed to novices by holding a modifier key down, but with a delay time to increase effort. In the extreme, costs can also be added to graphical input, like adding a delay after clicking on menu items [28], or even completely disabling menu item selection [38, 28].

One issue specific to keyboard shortcuts is poor visibility [28, 74]. Users often do not notice shortcuts shown on menu items or in tooltips. Proposed solutions reinforce shortcut information when graphical widgets are used, such as audio feedback when clicking menu items to remind users of shortcuts [28] or blending shortcut key characters into toolbar icons to increase their salience [22]. Our techniques do not make shortcuts more visible from existing graphical interfaces. Instead, we increase visibility by providing a guidance interface to keyboard shortcuts themselves.

Another issue with keyboard shortcuts is unintuitive command mappings [28, 34, 41, 61]. Due to practical constraints, few commands are mapped to the keys of their initial letters, often making mappings arbitrary. To increase alignment between command name and key, multiple modifier keys must be used, but this makes them difficult to memorise [87] and execute [41, 52, 74]. We address this by allowing more commands to be mapped to the same key using expressive hand postures as an alternative to multiple modifier keys.

Although there is significant work on learning keyboard shortcuts and several methods have been proposed to leverage novice to expert transitions, there has been no single study identifying the factors that hinders shortcut usage at a nuanced level. This observation motivates the interview study that follows.

STUDY 1: INTERVIEW WITH EXPERT USERS

We conducted an interview study with expert computer users to gain a deeper understanding of keyboard shortcut issues. Our results confirm, combine, and extend factors identified in previous literature across disparate studies, arguments, and informal observations. We use the factors to inform and motivate the design of Guided Finger-Aware Shortcuts, and these cohesive results form a useful resource for future researchers.

Participants

We recruited 12 participants (4 female), 21 to 39 years old (M=26.9, SD=5.9). Participants reported weekly computer usage from 42 to 84 hours (M=61.3, SD=10.6) and all had formal education or industry experience in Computer Science. They reported regular use of the Windows (7), macOS (7), and Linux (3) operating systems, and also experience with a variety of applications such as text-intensive applications (e.g. Google Docs, Microsoft Word, Sublime Text, Vim), graphics-intensive applications (e.g. Adobe Photoshop, GIMP, Blender, Autodesk 3ds Max), integrated development environments (e.g. Visual Studio, Xcode, Eclipse, Unity), and other common software (e.g. Google Chrome, Safari, Adobe Acrobat Reader, Skype). We recruited experts with a Computer Science background because they frequently perform repetitive tasks with high efficiency and can be reflective about challenges and strategies when learning shortcuts.

Procedure

Semi-structured interviews were conducted to probe the basic usage, learning, physical articulation, and currently available feedback of keyboard shortcuts (interview questions are in the Appendix). These aspects were formulated based on an analysis of the keyboard shortcut action using Norman's seven stages of action framework [56]. We reviewed the data to ensure saturation had occurred after 12 participants – the same patterns were recurring and additional information was unlikely to be obtained with more participants [19]. Thematic analysis [10] was employed to identify the themes within the data. Note that the interviewer's knowledge of the literature surrounding keyboard shortcuts could bias them towards previously identified issues. To mitigate this, the interviewer explicitly avoided asking leading questions related to known factors, and using a formal method like thematic analysis reduced analysis bias.

Results

The interview revealed nine factors that may impact the use of keyboard shortcuts. We report, compare, and contrast them with previous work in the discussion that follows.

Factor 1: Frequency of Use

Comments from participants suggest frequent activation of a command motivates the learning of a keyboard shortcut. This occurs naturally with repeated usage, not dedicated studying: "it's not like a dedicated learning process where I spend ten minutes just for memorising keyboard shortcuts" (*P*3). For the same reason, infrequent commands are likely to be forgotten: "I'd always pay attention to the keyboard shortcut while clicking on the menu ... but by the time I reuse the same command, I would already forget the shortcut" (*P*11).

Frequency of use also affects the development of automaticity. For very common actions such as *Copy*, *Paste*, and *Save*, all participants acknowledged that they have developed muscle memory – keyboard shortcuts can be articulated without much cognitive effort. In contrast, less frequent keyboard shortcuts require a conscious association to the keys being pressed: "for the things that I use less often like Bold in

Google Docs, I often have myself being like ... OK, I have to use Ctrl+B, so it's a little bit more thoughts into it" ($P5$).

Factor 2: Perceived Cost

Switching to keyboard shortcuts is often motivated by the inconvenience of graphical input: "... keyboard shortcuts [are] way more efficient than just clicking everywhere with a mouse" ($P3$). Indeed, users may unconsciously compare the perceived costs between using a mouse or keyboard. These costs, an estimation of each action's physical and cognitive effort, drive users to employ one method over another.

Different graphical widgets may require different extents of effort. In many situations, participants found it convenient enough to use context menus: "if I can do the task with a right click [and a context menu], I might not bother to remember the keyboard shortcut" ($P7$), or toolbars: "if it's just on the toolbar, I probably will have to use it a lot of times before I search for the shortcut" ($P10$). In these cases, the perceived cost of graphical input is relatively low, whereas menus are higher: "I feel it's more likely for me to use keyboard shortcuts if the commands are hidden in a menu" ($P9$).

The perceived costs are also related to the hand positions *before* and *after* the command action. When two hands are both on the keyboard, providing graphical input is costly: "... you have to move your hand back to the mouse, which is much more work than just using the shortcut" ($P3$). However, when one hand is already on the mouse, accessing graphical widgets becomes easier, which discourages the use of keyboard shortcuts ($P2,8,12$). If subsequent input is text entry, keyboard shortcuts are more likely to be used ($P11$); whereas if subsequent input uses a mouse, shortcuts are less likely: "keyboard shortcuts like Ctrl+P pops up a print settings dialogue – I have to use my mouse anyway" ($P7$).

Factor 3: Visibility

Lack of visibility is a major hindrance to learning keyboard shortcuts: "there's no real good way of discovering them" ($P5$). Because of this, participants adopt different strategies to find shortcuts including navigating menus ($P1-12$), hovering over toolbar icons ($P1-12$), performing menu search ($P10$), searching online ($P1-12$), studying crib sheets ($P1,2,7,10-12$), and watching tutorials ($P2,5,6$).

The most common ways of finding keyboard shortcuts are navigating menus and hovering on toolbar icons ($P1-12$). Given a certain command, users need to first locate the corresponding graphical element in the user interface. This is relatively easy if learning is prompted by frequent menu or toolbar access ($P10$), but not if they are unfamiliar with the user interface, despite having a goal in mind: "for something like Ctrl+K Ctrl+C for Comment in Visual Studio, I know the command must exist [from other code editors], but I'm not sure where to find it in the menu" ($P10$).

For the latter, some operating systems (e.g. macOS) and applications (e.g. Google Docs) feature the ability to search commands inside the menu ($P10$), but this was not well known among our participants. More participants resorted to online search, often with keywords containing the application name and a description of the action that the command performs ($P1-12$). Online search becomes more valuable for terminal applications, which heavily rely on keyboard shortcuts (e.g. Vim ($P6$)) or complex software with many functions (e.g. SolidWorks ($P6$)). In both cases, the visibility of keyboard shortcuts from the user interface alone is so low that users have to seek for external support.

Another common strategy is to use crib sheets ($P1,2,7,10-12$). Participants indicated crib sheets are most suitable for learning shortcuts they have used before: "if I've learned the shortcuts in the past, and I just need to quickly glance over them, then crib sheets are the best option" ($P10$). Some participants made their own crib sheets ($P6,12$), some searched online ($P1,2,7,10,11$), and others used ones provided by the software ($P9$). Many applications include the feature of quickly viewing a crib sheet with a keyboard shortcut, for example, "in Google Docs, you can use Command+/ to see all the shortcuts available" ($P10$). In addition to crib sheets, participants suggested watching online tutorials helps learn keyboard shortcuts, especially for feature-rich software like Visual Studio Code ($P5$), SolidWorks ($P6$), and Blender ($P11$).

Factor 4: Semantic Alignment to Associated Command

Participants commonly acknowledged the importance of semantic alignment between the shortcut key and its associated command ($P1-12$). Many suggested the most pronounced issue with keyboard shortcuts are that they "do not make sense" ($P1,2,4,6-12$). The misalignment with the associated command not only impedes learning, but also hinders the exploration of new keyboard shortcuts.

Participants indicated that keyboard shortcuts are most easily memorised when the key matches the first letter of a major word in the command (e.g. $\boxed{\text{Ctrl}}+\boxed{\text{S}}$ for *Save*, and $\boxed{\text{Ctrl}}+\boxed{\text{A}}$ for *Select All*) ($P1-12$). They might even carry this expectation to unknown shortcuts of common actions, for example: "sometimes I think Ctrl+D should be delete, and I'll do a Ctrl+D when I want to delete a line of code, but only to find out it actually does something else" ($P3$). When another alphabetical key is selected instead of the first letter of a command word (e.g. $\boxed{\text{Ctrl}}+\boxed{\text{G}}$ for *Find Next*), it is more likely to be confusing ($P1-12$): "Ctrl+S to save makes sense, but to strikethrough [text], for example, you have to try to figure out what the keyboard shortcut for that is, because the one that first comes to your mind is Ctrl+S" ($P6$).

Even worse than arbitrary alphabetical keys are modifier keys ($P1-12$): "anything that has to do with Ctrl, Alt, and Shift [is] very confusing: they are all in the same area and the combinations you need to hit in connection with another key is very confusing" ($P6$). Participants also suggested symbols used for modifier keys further aggravate the issue: "(pointing to a modifier key symbol on macOS) is this control [or] option? I'm always confused about these things" ($P8$).

Factor 5: Physical Articulation

Participants indicated the difficulty of pressing multiple keys also affects the use of keyboard shortcuts ($P1-12$). This can be attributed to both the cognitive complexity of memorising multiple keys ($P2-5,7-12$) and the physical complexity of articulating multiple keys $(P1, 5-7, 10)$. Most participants suggested that pressing three keys simultaneously with one hand is acceptable, but more keys may become hard to memorise or articulate $(P3-5, 7-12)$.

Participants preferred single-handed shortcuts $(P1, 3-7, 9, 11, 12)$. Having one hand on the mouse and the other activating keyboard shortcuts help with frequent mode switching when manipulating graphical objects, for example: "when I'm using Blender, I would typically have one hand on the keyboard to switch between different tools and the other hand on the mouse to manipulate objects" $(P10)$.

Some shortcuts require pressing keys in steps (e.g. Ctrl+K, Ctrl+C for *Comment* in Visual Studio). Participants found this hard to learn $(P4, 5, 11)$: "there're already too many things to remember in a keyboard shortcut, yet with these shortcuts, I'll also have to remember their ordering" $(P5)$.

Factor 6: Consistency
Comments from participants suggest the consistency of shortcut mappings across applications and operating systems helps them learn keyboard shortcuts. Participants described those that are retained to be "system-wide" $(P3, 5)$, "ubiquitous" $(P1, 6, 8)$, and "universal" $(P7, 9)$. These also include ones with obscure command mappings like ⌘+V for *Paste* $(P1, 2, 3, 5, 9)$, Ctrl+Alt+Del for *Windows Task Manager* $(P4)$, and F2 for *Rename* $(P12)$.

In contrast, keyboard shortcuts that participants tend to forget are typically specific to a certain application $(P1, 2, 4, 6, 7, 8, 9, 10, 12)$, especially ones with many shortcut commands like Adobe Photoshop $(P1)$, Adobe Illustrator $(P8)$, Inkscape $(P4)$, SolidWorks $(P6)$, Blender $(P11)$, and Vim $(P2, 3)$. Learning or relearning happens on an ad hoc basis based on current need: "I'd remember a keyboard shortcut for the day that I am using it, but the next day I might already forget it" $(P7)$.

Consistency motivates the exploration of new shortcuts: "sometimes I try the shortcut I know for other applications – I think 'well, this seems to be pretty standard, so maybe it works in this application too' " $(P4)$. In contrast, inconsistency impedes exploration: "I'd get confused when some standard shortcuts are mapped differently in some other applications" $(P4)$. Participants provided examples of this inconsistency: Ctrl+T does not open a new tab in Notepad++ $(P4)$, ⌘+Z does not perform an undo in Vim $(P8)$, and Ctrl+/ does not comment a line in Visual Studio $(P4, 11)$.

Factor 7: Perceived Risk of Making Errors
The perceived risk of making an error hinders shortcut use, especially for important tasks: "if I already have a three-page document full of important text, then I wouldn't try a keyboard shortcut that I'm not sure at all" $(P12)$. Many participants said they will use a keyboard shortcut only if very confident about its action $(P1, 3, 5, 7, 8, 10, 11, 12)$. Some participants felt that making an error is undesirable even if the action is non-destructive and correctable with undo $(P5, 7, 9, 11)$, maybe because the perceived cost of correcting the error is high. Others suggested they will experiment when shortcuts switch modes $(P2, 11)$, for example: "in 3ds Max, pressing number keys switches between editing vertices, edges, and faces; I know that the numbers switch between them, but I'm not quite sure, for example, whether two is the right number; in that case, I'd try pressing the keys until I see the one I want" $(P2)$.

Factor 8: Feedback
Participants suggested that feedback for the actions triggered by keyboard shortcuts also affects their adoption $(P1, 2, 5, 6, 8-12)$. Appropriate feedback helps confirm the action was correct: "... the interaction in SolidWorks is very effective, because you have immediate feedback on what's happened" $(P6)$, and "it would be easy to tell [what error I made] in Lightroom, because it pops up a toast whenever I trigger a shortcut" $(P10)$. Explicit feedback also encourages users to explore new keyboard shortcuts: "the toast notification in Lightroom also makes me more willing to experiment with keyboard shortcuts ... if I don't know what the error was, then I won't be able to learn from mistakes" $(P10)$.

Despite this, many participants noted poor shortcut feedback $(P1, 2, 5, 6, 8-12)$, especially when accidentally triggering the wrong action $(P1, 5, 6, 9-12)$. This is amplified when keyboard shortcuts trigger actions away from the user's focus: "knowing what command a keyboard shortcut has triggered is often very difficult unless it involves apparent UI changes; you won't know if you accidentally triggered a keyboard shortcut which does something in the background or toggles a small button on the toolbar" $(P11)$.

Factor 9: Customisation
A few participants suggested customised shortcut mappings compensate for the drawbacks of existing keyboard shortcuts and may be more easily memorised $(P4, 6, 10, 12)$, for example: "I typically remember quite well shortcuts I bound myself, so if I come up with a custom shortcut, I generally remember that much better than shortcuts that are given to me" $(P6)$.

Yet, they also indicated that customised shortcuts may be hard to communicate with other people $(P10)$ and difficult to find once forgotten $(P1)$. In addition, customisation typically requires more modifier keys to avoid conflicts with existing shortcuts: "when you customise a shortcut ... you have to use multiple modifier keys and try different combinations of modifier keys to see which one hasn't been used" $(P10)$.

Discussion
The study identified nine factors impacting the learning, use, and exploration of shortcut keys. We label them from $F1$ to $F9$ in the following discussion. Some of our observations are consistent with other work at a high level. For example, Kim and Ritter [37] also acknowledged frequency of use as a dominant factor in learning keyboard shortcuts $(F1)$; Grossman et al. [28] and Krisler and Alterman [38] found perceived cost $(F2)$, visibility $(F3)$, semantic alignment $(F4)$, and feedback $(F8)$ impact shortcut learning; Peres et al. [61] suggested perceived risk $(F7)$ also contributes to the underuse of keyboard shortcuts; Malacria [49] promoted keyboard shortcuts by increasing visibility $(F3)$ and decreasing the cost $(F2)$;

Pietrzak et al. [63] recognised the difficulty of pressing multiple modifier keys ($F5$) and duplicated modifier keys on a mouse; and Giannisakis et al. [22] encouraged shortcut usage by increasing the visibility on toolbar icons ($F3$). Our study connects and adds depth to the understanding of keyboard shortcuts at a more nuanced level and recognises other factors such as consistency ($F6$) and customisation ($F9$) that are not previously identified.

GUIDED FINGER-AWARE SHORTCUTS

We present details about two variations of Guided Finger-Aware Shortcuts, FingerArc and FingerChord, and explain how their design alleviates issues identified in the interview study. Note the techniques are complementary to menu interaction (and other graphical interfaces). They bridge menu navigation and direct shortcut activation with optional intermediate visual guidance, reducing the gulf between them (Fig. 2).

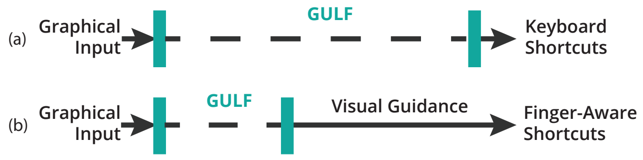

Figure 2: Illustration of FingerArc and FingerChord's interaction model compared to traditional keyboard shortcuts: (a) a huge gulf exists between graphical input and keyboard shortcuts; (b) FingerArc and FingerChord reduce the gulf by providing dynamic visual guidance when pressing and holding an action key.

There are two different modes for users at different levels of proficiency. *Guidance mode* is triggered when pausing on an alphabetic key (we call this the "action key") for a predesignated delay time with a special hand posture. A shortcut interface pops up after the delay. Subsequent adjustment of the hand posture selects from a group of commands. The interface provides dynamic visual feedback based on the current hand posture and eventually guides users to complete the action. *Shortcut mode* is triggered by pressing and releasing the action key with the hand posture that activates the command, and no visual guidance is shown.

FingerArc and FingerChord use different hand postures to differentiate shortcut activation from normal text entry (Fig. 3a). FingerArc expects the action key to be pressed using the index finger with the little, ring, and middle fingers tucked in. FingerChord expects the middle finger to press the action key and the little and ring fingers tucked in. Like Finger-Aware Shortcuts [87], hand postures can be detected using computer vision with a camera facing down mounted on top of the screen. Using two different fingers to initiate the action makes the two techniques compatible with each other, and the special hand postures make them compatible with conventional keyboard shortcuts.

Both variations use other finger movements to select from a group of commands associated with the action key (Fig. 3b-e). FingerArc selects the primary command by folding all fingers except the index finger used for pressing the key. Other options are specified based on the angle between the index finger and the thumb: 90° selects the secondary command, 45° selects the tertiary command, and 0° selects the quaternary command. FingerChord selects the primary command by default if no other key is pressed. Other options are specified using a second key press using the index finger on different keyboard rows relative to the action key. The lower row selects the secondary command, middle row selects the tertiary command, and upper row selects the quaternary command.

The selected command is activated for both FingerArc and FingerChord by releasing all keys while maintaining the hand posture (Fig. 3f). If two keys are pressed in FingerChord, they need to be released simultaneously (within 200ms). In both techniques, cancellation without triggering any command is achieved by opening all the fingers before releasing the key (Fig. 3g).

The techniques include six design features to address issues identified in the interview study and facilitate novice to expert transitions:

Expressivity — FingerArc and FingerChord increase the number of commands associated with a single key by eight, without requiring inconsistent articulation of multiple modifier keys. A larger expressive space allows for increased semantic alignment ($F4$) because the same action key can be used for more commands, and this also provides opportunities for increased shortcut customisation ($F9$). Increased expressivity also creates more input space to make shortcut keys more consistent across applications ($F6$).

Rehearsal — The principle of rehearsal is embodied by providing an interface for practising and refining shortcut gestures. The gap between graphical input and shortcut activation is shortened since users now have an opportunity to "play with" the shortcuts before mastering them.

Guidance — The techniques provide dynamic visual guidance to enable rehearsal of shortcut keys, making them easier to use ($F2$). The guidance shows the available shortcuts for an action key to increase the visibility of shortcut commands ($F3$), and provides feedback about what command is currently selected ($F8$) to reduce the perceived risk of making an error ($F7$).

Effort — Adding a small time delay before triggering the shortcut guidance slightly increases the effort, therefore subtly nudging users towards activating shortcuts directly.

Risk Mitigation — Either technique variation can be initiated to show the guidance feedback, then cancelled without selecting a command. This reduces the perception of risk ($F7$) and encourages exploration using the novice mode.

Consistent Articulation — Both techniques use consistent finger positioning regardless of the action key location, which simplifies articulation ($F5$). This is different from conventional shortcuts that often require pressing one or more modifiers with different distances from the action key.

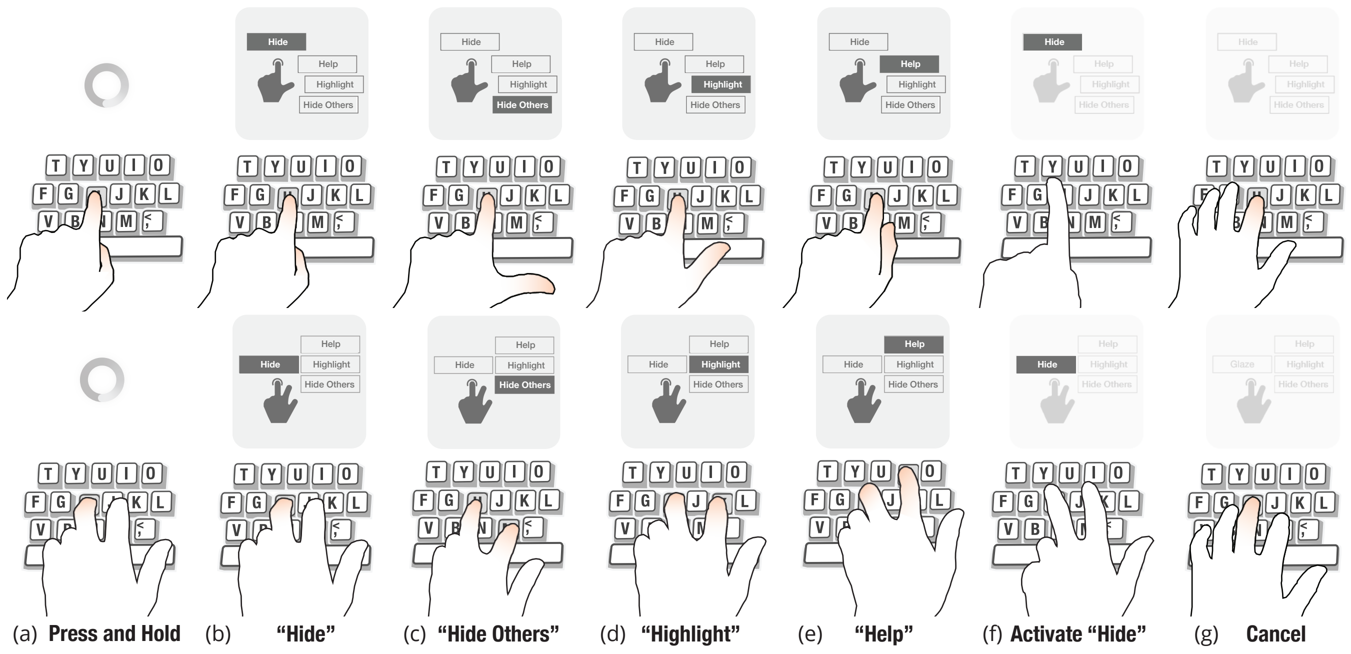

Figure 3: FingerArc and FingerChord: (a) holding an action key with a special hand posture for a predesignated delay time reveals the shortcut interface; (b) selecting the primary command using the index finger with others tucked in (FingerArc) or the middle finger (FingerChord); (c-e) selecting other commands using the angle of the thumb (FingerArc) or pressing different key areas (FingerChord); (f) releasing the key maintaining a hand posture triggers the command (e.g. primary command); (g) revealing all the fingers while holding the key cancels the operation.

STUDY 2: CONTROLLED EXPERIMENT

The objective of this study is to assess the performance of Guided Finger-Aware Shortcuts in supporting novice to expert transitions. We compare FingerArc and FingerChord with a standard keyboard shortcut baseline using modifier keys ("Baseline") and a condition that provides guidance for standard shortcuts ("GuidedKey"). GuidedKey allows holding down ⌘ with an action key to reveal visual guidance, then incrementally adjusting the modifier keys with visual feedback. A command is activated by releasing all keys together, or cancelled by releasing ⌘ first. GuidedKey models interaction techniques like ExposeHK [49] that provide visual guidance for standard shortcuts, and also provides a similar mechanism for cancellation like FingerArc and FingerChord. Including GuidedKey helps generalise the role of visual guidance in promoting novice to expert transitions and to discern the benefits of making shortcuts finger-aware.

Participants and Apparatus

We recruited 32 participants (5 female, 3 left-handed but all used their right hand to operate a touchpad), ages ranging from 20 to 50 years (M=25.4, SD=5.3). All reported extensive computer usage with weekly computer use time ranging from 28 to 112 hours (M=64.3, SD=20.6). A one-minute typing test revealed an average speed of 52.8 words per minute (SD=17.8).

The experiment was conducted on a MacBook Pro laptop with a 15-inch display and a QWERTY keyboard. Similar to Finger-Aware Shortcuts [87], we also used computer vision to detect hand postures. A downward facing camera fixed in a 3D printed case was mounted on top of the laptop screen, recording the keyboard area. The camera was equipped with an illumination board and an infrared filter to control the lighting environment.

Since our focus is on interaction techniques, not tracking algorithms, we used markers to facilitate hand tracking. Three 5mm hemisphere reflective markers were attached to the fingertips of the little finger, index finger, and the thumb, respectively. A fourth marker was attached to a point on the back of the hand that forms a right angle with the markers on the index finger and the thumb.

A native macOS application ran as a system-wide service, tracking the marker positions while intercepting keyboard events. Keyboard area was rectified by manually labelling the four corners of the keyboard in the software. Heuristic rules were applied to identify hand postures in real-time. The experiment interface was implemented as a fullscreen web application in Google Chrome.

Task and Stimuli

Our task was adapted from Grossman et al. [28] and Appert and Zhai [2], both examining standard shortcut learning. We also included reference tasks after Zheng and Vogel [87] so command selection is interspersed with other tasks. A trial in our experiment required first completing a reference task, then immediately a command task.

Reference Task

There were two reference tasks: *typing* and *pointing*. The *typing task* required entering three keys on the keyboard home row to position both hands on the keyboard. The *pointing task* required clicking on a button at the centre of the screen to position the dominant pointing hand on the touchpad.

Command Task

In the command task, the participant was prompted to select a five-letter command name appearing at the centre of the screen. The participant could use a standard application linear menu at the top of the screen, or a keyboard shortcut (if available for the stimulus command). The keyboard shortcut could only be performed with the keyboard shortcut technique condition. If the participant triggered a wrong command in the command task, the reference task was repeated followed by the same command task until successfully activated. Therefore, a trial may consist of one or more attempts to activate the prompted command.

Shortcut Techniques

We tested 4 shortcut conditions: FingerArc, FingerChord, GuidedKey, and Baseline. The 4 conditions all overload 4 commands to an action key, but differentiate the commands using different overloading methods and trigger shortcuts on different key events. We summarise these in Table 1. The 4 postures in each technique were informally tested and ranked by ease of physical articulation by the authors, so that the primary postures are associated with the most frequent commands, and quaternary postures are assigned to infrequent commands. At the end of the experiment, we elicited ratings for these postures to help refine our techniques.

	Baseline	GuidedKey	FingerArc	FingerChord
Overloading Method	modifier key	modifier key	thumb to index angle	secondary key press
Trigger Event	key down	key up	key up	key up
Primary Posture	⌘	⌘	thumb hidden	no secondary key press
Secondary Posture	⌘+shift	⌘+shift	90°	lower row
Tertiary Posture	⌘+ctrl	⌘+ctrl	45°	same row
Quaternary Posture	⌘+alt	⌘+alt	0°	upper row

Table 1: Methods for overloading the same key press, trigger events of shortcuts, and postures for activating four commands bound with a key.

Menu Interface

There were 60 commands, divided into 12 sets of 5 commands, each set starting with a different letter. Half of the 12 letters were randomly chosen from the left side of the keyboard, we call *left keys* (e.g. C, S, T, G, R, Q), the other half from the right side, we call *right keys* (e.g. P, Y, U, J, I, N). In each set, 4 commands were assigned with keyboard shortcuts using the 4 postures described above. We call these commands *primary command*, *secondary command*, *tertiary command*, and *quaternary command*, corresponding to the postures. The fifth command in the set was only accessible from the menu with no shortcut assigned. Keyboard shortcuts in the menu were displayed as symbols similar to the shortcut guidance for FingerArc and FingerChord (Fig. 4), and as plain modifier text (e.g. Command + Shift + C) for GuidedKey and Baseline.

An application menu at the top of the screen (Fig. 4) arranged the 60 commands into 6 menus, each labelled with a letter chosen from the 12 letters above (e.g. S, P, C, Y, T, U). The labels alternated between using left keys and right

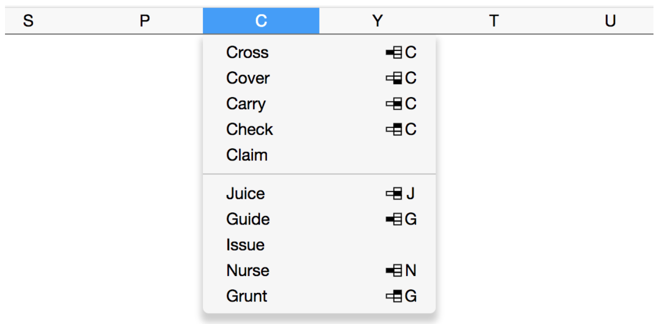

Figure 4: Experiment Menu interface: 'Cross', 'Cover', 'Carry', 'Check', and 'Claim' are a *grouped* set since they appear together in the 'C' menu; 'Claim' is an *unmappped* distractor; the other 5 commands belong to *ungrouped* sets; 'Issue' is another *unmappped* distractor.

keys. Each menu contained 10 commands, visually divided into two sets of 5 using a line separator. Half of the commands were *grouped*, meaning they all start with the same letter as the menu label. For example, 'Cross', 'Cover', 'Carry', 'Check', and 'Claim' were all grouped together under the 'C' menu (Fig. 4). The other 5 commands in the menu had different first letters. The grouped commands alternated between showing in the first half and the last half in the menu. The grouped commands were ordered so that the primary command appeared at the top and the one with no shortcut showed at the bottom.

Note that *grouped* commands capture shortcuts with good semantic alignment, something the preceding interview identified as making memorising keyboard shortcuts easier. However, most of our analysis focuses on *ungrouped* commands since our techniques are designed for shortcuts that are typically harder to remember.

Command Stimuli

Although the menu displayed 60 commands in total, only 20 were used as the stimuli for the experiment tasks. The other 40 served as passive *distractors* to fill the menus and make searching for commands more realistic. The 20 commands were from 4 sets of the 12 command sets described above, each set using a different first letter. The 4 sets were selected to balance the left and right keys. In addition, two of these sets were *grouped*, each appearing under a menu named with the first letter used by the set. This simulates an ideal case where users can see immediately where the command is located in the menu. The other two sets were *ungrouped*, meaning they were distributed across other menus. This models searching for unfamiliar commands.

Of the 20 commands, 4 were *unmapped*, meaning they were not mapped to shortcuts. This models real applications, where not every command has a shortcut. These served as another type of distractor, and also forced the participant's attention back to the application menu similar to what happens in real usage. The remaining 16 commands were all *mapped* to shortcuts, so the participant could utilise the shortcut technique and memorise them. Our analysis only looks at factors relating to these 16 mapped commands.

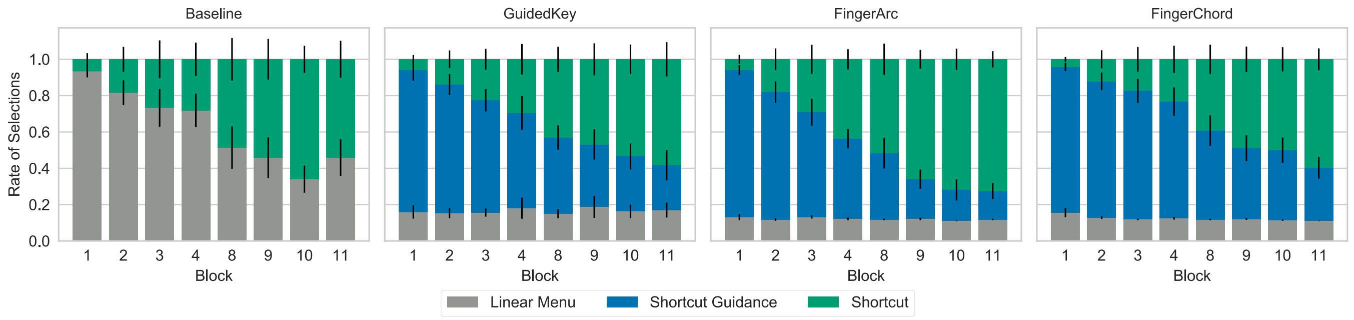

Figure 5: Modalities (MENU, GUIDANCE, SHORTCUT) used over training blocks for each shortcut technique.

Frequency

The frequency each command appeared during a block followed the tail of a Zipfian distribution, each of the 20 commands appeared this many times per block: 3, 3, 3, 3, 2, 2, 2, 2, 2, 2, 2, 2, 1, 1, 1, 1, 1, 1, 1, 1. This simulates the usage of less frequent commands in real applications [81, 48]. We removed the first 8 items from the complete Zipfian distribution (30, 15, 10, 8, 6, 5, 4, 4) because our focus is to improve the learning of less frequent commands, not frequent ones like ⌘+C or ⌘+S that people typically know. We assigned the four primary commands with frequency 3, four secondary commands with frequency 2, four tertiary commands with frequency 2, four quaternary commands with frequency 1, and four commands with no shortcut with frequency 1.

Design

Our experiment is a mixed design. We used a between-subject factor for the 4 shortcut TECHNIQUES, each tested with 8 participants. Command GROUPING and BLOCK form two within-subject factors. The experiment consisted of multiple blocks over two days. After demonstrating the assigned technique, the participant was presented with 4 *Training blocks* (BLOCKS 1 - 4), followed by a *Test block* (BLOCK 5), followed by a five-minute distraction task where the participant switched focus to play a breakout game on a phone (BLOCK 6), followed by another *Test block* (BLOCK 7). Then, the participant practised for another 4 *Training blocks* (BLOCKS 8 - 11), followed by a *Test block* (BLOCK 12). Finally, after 24 hours, the participant completed a final *Test block* (BLOCK 13). *Training blocks* alternated between typing and pointing reference tasks, and the order was counterbalanced across participants. In all *Test blocks*, only the 16 commands assigned with shortcuts were presented. The participant needed to invoke the commands using only the given shortcut technique. Only one attempt was allowed, and the main menu was hidden and shortcut guidance was disabled. Participants were instructed to complete the tasks as quickly and accurately as possible, and they were not told about *Test blocks* beforehand to prevent an explicit focus on memorisation.

Results

A successful novice to expert transition can be characterised from three different aspects over time: a modality switch from the novice to expert method, increased motor performance as measured by time, increased memory performance as measured by successful expert method usage during *Training blocks* and shortcut recall in *Test blocks*. Errors made during *Training blocks* may provide evidence for both motor and memory performance, though determining the exact source of error is not possible in the current study. We examine these metrics in the following analysis.

Modality Change

During training, there were three modalities to trigger a command: menu interface (MENU), shortcut with guidance (GUIDANCE), and direct shortcut without guidance (SHORTCUT). We examined the modality change in *Training blocks* for each shortcut technique. Fig. 5 illustrates similar curves for SHORTCUT usage (green bars) for all techniques, but for those with guidance, GUIDANCE usage was very high in BLOCK 1 then gradually decreased in the following blocks. The MENU usage for these techniques was constantly very low (grey bars) compared to Baseline.

The result suggests that guided techniques help users switch to using keyboard shortcuts early, then gradually transition from relying on guidance to memory recall. Early adoption of shortcut guidance may be due to the novelty effect, but we found participants persisted with keyboard shortcuts rather than reverting back to the menu interface. This implies certain benefits to providing shortcut guidance, which we examine in more detail in the quantitative analysis that follows. Note for all subsequent analysis, trials are aggregated by participant and the factors being analysed. Time data were aggregated using median.

Learning Effect

We analysed learning effect by examining command task time of trials in *Training blocks*. Time for reference tasks was excluded since it is irrelevant to command activation. In cases where the participant made an error and had to repeat the trial, we sum the time for invoking the command across all attempts. Time was not normally distributed and as such we performed a log transformation.

We found a main effect of BLOCK with a one-way repeated measures ANOVA ($F_{3.09,95.94} = 29.84$, $p < .001$, $\eta^2_G = .27$)[1], and post hoc tests suggest learning flattened from BLOCK 8 through 11 (all $p > .05$) as seen in Fig. 6. These blocks provide the most stable measurement of time, error, and expert use when the performance in the training blocks had levelled

[1]All standardised effect sizes are generalised eta squared [60, 5]

out. In subsequent analysis of *Training blocks*, we only use blocks 8 to 11.

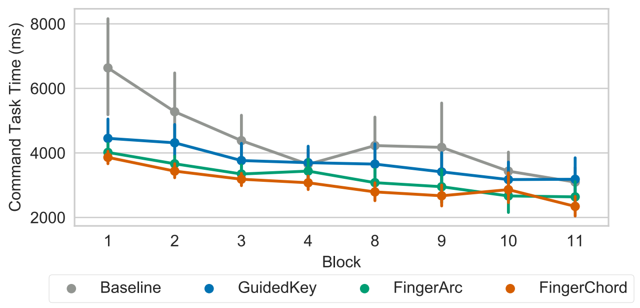

Figure 6: Command task time by *Training block*.

Time

Fig. 7 illustrates the overall command task time in *Training blocks*. Splitting by modality showed average times of 6699ms (SD=2241) for MENU, 4450ms (SD=1348) for GUIDANCE, and 2029ms (SD=854) for SHORTCUT, clearly demonstrating the invaluable performance benefit of transitioning to keyboard shortcuts. Time was not normally distributed and as such we log transformed the data. A mixed factorial ANOVA of BLOCK × GROUPING × TECHNIQUE revealed an interaction between TECHNIQUE × GROUPING ($F_{3,28} = 3.00$, $p < 0.05$, $\eta^2_G = 0.03$). We performed post hoc analysis using pairwise t-tests and manually corrected for multiple comparison using the Bonferroni method. For UNGROUPED commands, our analysis showed FingerArc outperformed Baseline by 1162ms and GuidedKey by 731ms, and FingerChord outperformed Baseline by 1169ms and GuidedKey by 738ms (all $p < 0.05$). For GROUPED commands, FingerChord outperformed GuidedKey by 486ms ($p < 0.01$). The time difference suggests better motor performance for overall command invocation with finger-aware techniques. Although a standardised effect size of 0.03 is considered small [5], the size of the simple effect for UNGROUPED commands, a reduction of more than 1.2 seconds for FingerArc and FingerChord compared to Baseline, would be noticeable to users.

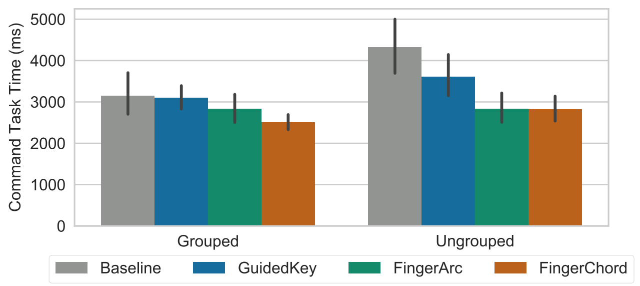

Figure 7: Command task time by command grouping type.

Expert Use

Fig. 8 illustrates successful expert use in *Training blocks*. Expert use is an indicator variable that is true when a trial was completed in one attempt using a shortcut without guidance. A mixed factorial ANOVA of BLOCK × GROUPING × TECHNIQUE showed no interaction effect of TECHNIQUE × GROUPING justifying separate analysis of GROUPED and UNGROUPED ($F_{3,28} = 2.54$, $p = 0.08$, $\eta^2_G = 0.009$) commands. Two one-way between-subjects ANOVAs for GROUPED and UNGROUPED commands revealed main effects of TECHNIQUE on expert use for both GROUPED ($F_{3,124} = 2.8$, $p < .05$, $\eta^2_G = .07$) and UNGROUPED ($F_{3,124} = 6.89$, $p < .001$, $\eta^2_G = .14$) commands. For GROUPED commands, post hoc tests using Tukey's HSD showed FingerArc had 14.8% more expert use than GuidedKey ($p < .05$). For UNGROUPED commands, FingerArc had 15.1% to 19.2% more expert use than FingerChord, GuidedKey, and Baseline (all $p < 0.05$). Although the standardised effect size of 0.14 for UNGROUPED is small [5], the simple effect size leading to 19% more expert selections for FingerArc compared to Baseline means users use one more shortcut with FingerArc for every five they would use with current shortcuts.

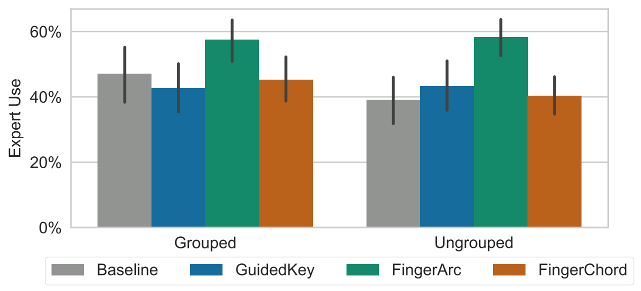

Figure 8: Rate of expert use.

Errors

Fig. 9 shows the trial-level error rates in *Training blocks*. An error trial is when a participant failed to trigger the correct command in the first attempt, regardless of the modality used. An analysis of the attempts that use the menu revealed a very low 0.4% error rate (SD=6.1%). This means the main source of error was shortcut activation.

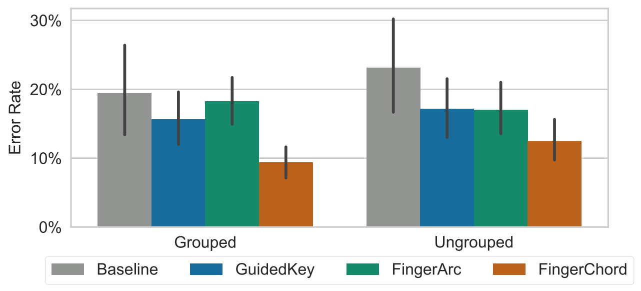

Figure 9: Error rate.

Again for errors we performed a mixed factorial ANOVA of BLOCK × GROUPING × TECHNIQUE which showed no interaction effect of TECHNIQUE × GROUPING justifying separate analysis of GROUPED and UNGROUPED commands ($F_{3,28} = 1.33$, $p = 0.28$, $\eta^2_G = 0.001$). Separate one-way ANOVAs found main effects of TECHNIQUE on error for both GROUPED ($F_{3,124} = 3.98$, $p < .01$, $\eta^2_G = .09$) and UNGROUPED ($F_{3,124} = 2.95$, $p < .05$, $\eta^2_G = .07$) commands. Tukey's HSD suggests for GROUPED commands, FingerChord had fewer errors than Baseline by 10.1% and FingerArc by 8.9% (both $p < .05$). For UNGROUPED commands, FingerChord had 10.6% less errors than Baseline ($p < .05$). Again the standardised effect sizes of .09 and .07 are small [5], but the simple effect sizes suggest selections with FingerChord are twice as likely to be correct compared to current shortcuts. FingerChord makes overall command invocation less error-prone, and FingerArc has comparable performance to existing techniques.

On average, participants made 1.2 attempts (SD=0.56) in a trial to invoke the command, suggesting a tendency of minimising errors. An analysis of the cancellation gestures revealed that 7.8% of the trials (SD=26.8%) had cancellation, showing participants indeed used this mechanism to explore shortcuts and avoid triggering undesirable commands.

Recall
After every four blocks of training, we performed two recall tests. The patterns in recall were similar before and after the distraction task, and before and after the 24-hour break. We report only the test after the 5-minute distraction task (BLOCK 7) and the test after the 24-hour break (BLOCK 13).

A mixed factorial ANOVA of GROUPING × TECHNIQUE showed no interaction effect justifying separate analysis of GROUPED and UNGROUPED commands ($F_{3,28} = 0.86$, $p = 0.47$, $\eta^2_G = 0.026$). We used separate one-way ANOVAs for GROUPED and UNGROUPED for the two recall blocks. In *Test block* 7, we only found a main effect of TECHNIQUES on recall for GROUPED commands ($F_{3,28} = 3.067$, $p < .05$, $\eta^2_G = .25$). Tukey's HSD showed a near significant difference between FingerArc and Baseline ($p = 0.053$). Baseline had fewer GROUPED commands recalled, maybe because selecting GROUPED commands in the menu was fast and easy that the participants did not pay attention to the shortcuts. This changed in BLOCK 13 probably due a testing effect.

In *Test block* 13, we found no main effect for either GROUPED or UNGROUPED commands, suggesting similar memory performance for all techniques. There could be a ceiling effect given the small command set, and this may have prevented significant differences in the 24-hour retention test.

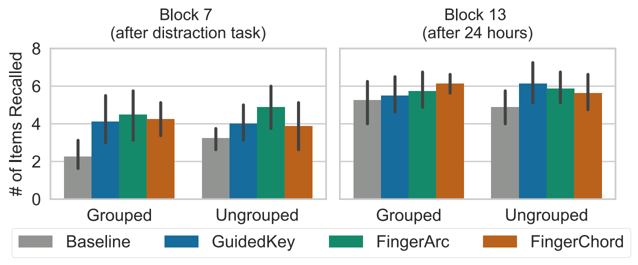

Figure 10: Number of items recalled in *Test block* 7 following the distractor task and in *Test block* 13 after 24 hours.

Subjective Ratings
At the end of the experiment, we elicited subjective ratings for each technique. There were no significant differences for TECHNIQUE in any subjective rating. However, two ratings are most interesting. Participants rated ease-of-memorisation to be quite consistent across all techniques (all medians between 2.0 and 3.0) in spite of measurable gains in the objective data. In addition, participants did not report a measurable difference in hand fatigue across techniques (all medians between 2.5 and 4.0).

We also elicited posture ratings for each technique as displayed in Table 2. The ratings showed that our informal rankings based on ease of physical articulation was reasonable.

Posture	**Baseline**	**GuidedKey**	**FingerArc**	**FingerChord**
Primary	5.0 (0.0)	5.0 (0.0)	4.5 (0.5)	5.0 (0.0)
Secondary	4.0 (0.5)	4.0 (0.5)	4.0 (0.5)	5.0 (0.0)
Tertiary	3.0 (0.0)	3.0 (1.0)	2.0 (1.0)	4.5 (0.5)
Quaternary	2.0 (1.0)	2.0 (0.5)	1.0 (0.0)	2.5 (0.5)

Table 2: Subjective ratings of postures within each technique reported as medians and median absolute deviations (MAD) in parentheses.

Discussion
Our experiment tested the novice to expert transitions of four shortcut conditions. The evaluation showed that guidance and finger-awareness mechanisms work together to increase both memory and motor performance when executing commands.

In terms of memory performance, users recalled a similar number of commands for all techniques, probably due to a ceiling effect from the small command set (Fig. 10). However, we observed early and persistent adoption of shortcut guidance for FingerArc, FingerChord, and GuidedKey (Fig. 5). Users smoothly transitioned from visually-guided shortcut selection to recall-based activation for these techniques. Additionally, FingerArc users made more shortcut invocations, and transitioned away from guided techniques more rapidly than other techniques.

In terms of motor performance, adoption of keyboard shortcuts resulted in improved command task times (Fig. 7). FingerArc and FingerChord were both faster than Baseline for ungrouped commands, suggesting that consistent articulation of commands successfully makes them easier to invoke when multiple modifiers are required for a traditional shortcut interface. Additionally, we show that users of guided techniques successfully utilised cancellation to avoid errors, suggesting a lower perceived risk of making errors.

CONCLUSION
We designed, implemented, and evaluated two variations of Guided Finger-Aware Shortcuts to alleviate the issues that hinder the adoption of keyboard shortcuts. By identifying the finger used to press a key, and using secondary finger movements to select commands, the two compatible techniques increase the number of commands associated with a key by eight. By introducing dynamic visual guidance to guide the user through the shortcut action, we made shortcuts easier to learn, use, and explore. Our evaluations showed that our techniques encourage overall shortcut usage, make interaction faster, less error-prone, and provide advantages over simply adding visual guidance to standard shortcuts. Our findings are promising, and to actually deploy our techniques, there is still much work to be done. For example, robust sensing algorithm needs to be developed to reduce markers, studies may be conducted to examine the persistence of shortcut usage in real-world settings, and further evaluation may reveal the long-term costs and benefits of our techniques.

REFERENCES

1. Fraser Anderson and Walter F. Bischof. 2013. Learning and performance with gesture guides. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, 1109–1118. http://dl.acm.org/citation.cfm?id=2466143

2. 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 (CHI '09)*. ACM, New York, NY, USA, 2289–2298. DOI: http://dx.doi.org/10.1145/1518701.1519052

3. Jeff Avery, Mark Choi, Daniel Vogel, and Edward Lank. 2014. Pinch-to-zoom-plus: An Enhanced Pinch-to-zoom That Reduces Clutching and Panning. In *Proceedings of the 27th Annual ACM Symposium on User Interface Software and Technology (UIST '14)*. ACM, New York, NY, USA, 595–604. DOI: http://dx.doi.org/10.1145/2642918.2647352

4. Gilles Bailly, Eric Lecolinet, and Yves Guiard. 2010. Finger-count & Radial-stroke Shortcuts: 2 Techniques for Augmenting Linear Menus on Multi-touch Surfaces. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '10)*. ACM, New York, NY, USA, 591–594. DOI: http://dx.doi.org/10.1145/1753326.1753414

5. Roger Bakeman. 2005. Recommended effect size statistics for repeated measures designs. *Behavior Research Methods* 37, 3 (Aug. 2005), 379–384. DOI: http://dx.doi.org/10.3758/BF03192707

6. 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 (UIST '08)*. ACM, New York, NY, USA, 37–46. DOI: http://dx.doi.org/10.1145/1449715.1449724

7. Suresh K. Bhavnani and Bonnie E. John. 1997. From Sufficient to Efficient Usage: An Analysis of Strategic Knowledge. In *Proceedings of the ACM SIGCHI Conference on Human Factors in Computing Systems (CHI '97)*. ACM, New York, NY, USA, 91–98. DOI: http://dx.doi.org/10.1145/258549.258615

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

9. Sebastian Boring, David Ledo, Xiang 'Anthony' Chen, Nicolai Marquardt, Anthony Tang, and Saul Greenberg. 2012. The Fat Thumb: Using the Thumb's Contact Size for Single-handed Mobile Interaction. In *Proceedings of the 14th International Conference on Human-computer Interaction with Mobile Devices and Services (MobileHCI '12)*. ACM, New York, NY, USA, 39–48. DOI:http://dx.doi.org/10.1145/2371574.2371582

10. Virginia Braun and Victoria Clarke. 2006. Using thematic analysis in psychology. *Qualitative Research in Psychology* 3, 2 (Jan. 2006), 77–101. DOI: http://dx.doi.org/10.1191/1478088706qp063oa

11. Daniel Buschek, Bianka Roppelt, and Florian Alt. 2018. Extending Keyboard Shortcuts with Arm and Wrist Rotation Gestures. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems (CHI '18)*. ACM, New York, NY, USA, 21:1–21:12. DOI:http://dx.doi.org/10.1145/3173574.3173595

12. John M. Carroll and Mary Beth Rosson. 1987. *Paradox of the active user.* The MIT Press. http://psycnet.apa.org/psycinfo/1987-98055-005

13. Andy Cockburn, Carl Gutwin, Joey Scarr, and Sylvain Malacria. 2014. Supporting Novice to Expert Transitions in User Interfaces. *ACM Comput. Surv.* 47, 2 (Nov. 2014), 31:1–31:36. DOI: http://dx.doi.org/10.1145/2659796

14. 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 (CHI '07)*. ACM, New York, NY, USA, 1571–1580. DOI: http://dx.doi.org/10.1145/1240624.1240863

15. Ashley Colley and Jonna Häkkilä. 2014. Exploring Finger Specific Touch Screen Interaction for Mobile Phone User Interfaces. In *Proceedings of the 26th Australian Computer-Human Interaction Conference on Designing Futures: The Future of Design (OzCHI '14)*. ACM, New York, NY, USA, 539–548. DOI: http://dx.doi.org/10.1145/2686612.2686699

16. Paul H. Dietz, Benjamin Eidelson, Jonathan Westhues, and Steven Bathiche. 2009. A Practical Pressure Sensitive Computer Keyboard. In *Proceedings of the 22Nd Annual ACM Symposium on User Interface Software and Technology (UIST '09)*. ACM, New York, NY, USA, 55–58. DOI: http://dx.doi.org/10.1145/1622176.1622187

17. Brian D. Ehret. 2002. Learning Where to Look: Location Learning in Graphical User Interfaces. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '02)*. ACM, New York, NY, USA, 211–218. DOI: http://dx.doi.org/10.1145/503376.503414

18. Bruno Fruchard, Eric Lecolinet, and Olivier Chapuis. 2017. MarkPad: Augmenting Touchpads for Command Selection. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems (CHI '17)*. ACM, New York, NY, USA, 5630–5642. DOI: http://dx.doi.org/10.1145/3025453.3025486

19. Patricia I. Fusch and Lawrence R. Ness. 2015. Are We There Yet? Data Saturation in Qualitative Research. *The Qualitative Report; Fort Lauderdale* 20, 9 (Sept. 2015), 1408–1416. https://search.proquest.com/docview/1721368991/abstract/2B4BDE44AC7B4DCCPQ/1

20. Varun Gaur, Md. Sami Uddin, and Carl Gutwin. 2018. Multiplexing Spatial Memory: Increasing the Capacity of FastTap Menus with Multiple Tabs. In *MobileHCI '18: 20th International Conference on Human-Computer Interaction with Mobile Devices and Services*. Barcelona, Spain, 13 pages.

21. Emilien Ghomi, Stéphane Huot, Olivier Bau, Michel Beaudouin-Lafon, and Wendy E. Mackay. 2013. ArpèGe: Learning Multitouch Chord Gestures Vocabularies. In *Proceedings of the 2013 ACM International Conference on Interactive Tabletops and Surfaces (ITS '13)*. ACM, New York, NY, USA, 209–218. DOI: http://dx.doi.org/10.1145/2512349.2512795

22. Emmanouil Giannisakis, Gilles Bailly, Sylvain Malacria, and Fanny Chevalier. 2017. IconHK: Using Toolbar Button Icons to Communicate Keyboard Shortcuts. 12. DOI: http://dx.doi.org/10.1145/3025453.3025595

23. Alix Goguey, Géry Casiez, Thomas Pietrzak, Daniel Vogel, and Nicolas Roussel. 2014. Adoiraccourcix: Multi-touch Command Selection Using Finger Identification. In *Proceedings of the 26th Conference on L'Interaction Homme-Machine (IHM '14)*. ACM, New York, NY, USA, 28–37. DOI: http://dx.doi.org/10.1145/2670444.2670446

24. Alix Goguey, Géry Casiez, Daniel Vogel, and Carl Gutwin. 2018. Characterizing Finger Pitch and Roll Orientation During Atomic Touch Actions. In *CHI 2018 - ACM Conference on Human Factors in Computing Systems*. Montréal, Canada, 1–12. DOI: http://dx.doi.org/10.1145/3173574.3174163

25. Alix Goguey, Mathieu Nancel, Géry Casiez, and Daniel Vogel. 2016. The Performance and Preference of Different Fingers and Chords for Pointing, Dragging, and Object Transformation. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems (CHI '16)*. ACM, New York, NY, USA, 4250–4261. DOI: http://dx.doi.org/10.1145/2858036.2858194

26. Alix Goguey, Daniel Vogel, Fanny Chevalier, Thomas Pietrzak, Nicolas Roussel, and Géry Casiez. 2017. Leveraging finger identification to integrate multi-touch command selection and parameter manipulation. *International Journal of Human-Computer Studies* 99 (March 2017), 21–36. DOI: http://dx.doi.org/10.1016/j.ijhcs.2016.11.002

27. D. O. Gorodnichy and A. Yogeswaran. 2006. Detection and tracking of pianist hands and fingers. In *The 3rd Canadian Conference on Computer and Robot Vision (CRV'06)*. 63–63. DOI: http://dx.doi.org/10.1109/CRV.2006.26

28. 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 (CHI '07)*. ACM, New York, NY, USA, 1591–1600. DOI: http://dx.doi.org/10.1145/1240624.1240865

29. Aakar Gupta, Muhammed Anwar, and Ravin Balakrishnan. 2016. Porous Interfaces for Small Screen Multitasking Using Finger Identification. In *Proceedings of the 29th Annual Symposium on User Interface Software and Technology (UIST '16)*. ACM, New York, NY, USA, 145–156. DOI: http://dx.doi.org/10.1145/2984511.2984557

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 (GI '15)*. Canadian Information Processing Society, Toronto, Ont., Canada, Canada, 223–231. http://dl.acm.org/citation.cfm?id=2788890.2788930

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 32Nd Annual ACM Conference on Human Factors in Computing Systems (CHI '14)*. ACM, New York, NY, USA, 2617–2626. DOI: http://dx.doi.org/10.1145/2556288.2557136

32. Chris Harrison and Scott Hudson. 2012. Using Shear As a Supplemental Two-dimensional Input Channel for Rich Touchscreen Interaction. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '12)*. ACM, New York, NY, USA, 3149–3152. DOI: http://dx.doi.org/10.1145/2207676.2208730

33. Chris Harrison, Julia Schwarz, and Scott E. Hudson. 2011. TapSense: Enhancing Finger Interaction on Touch Surfaces. In *Proceedings of the 24th Annual ACM Symposium on User Interface Software and Technology (UIST '11)*. ACM, New York, NY, USA, 627–636. DOI: http://dx.doi.org/10.1145/2047196.2047279

34. Jeff Hendy, Kellogg S. Booth, and Joanna McGrenere. 2010. Graphically Enhanced Keyboard Accelerators for GUIs. In *Proceedings of Graphics Interface 2010 (GI '10)*. Canadian Information Processing Society, Toronto, Ont., Canada, Canada, 3–10. http://dl.acm.org/citation.cfm?id=1839214.1839217

35. Seongkook Heo, Jingun Jung, and Geehyuk Lee. 2016. MelodicTap: Fingering Hotkey for Touch Tablets. In *Proceedings of the 28th Australian Conference on Computer-Human Interaction (OzCHI '16)*. ACM, New York, NY, USA, 396–400. DOI: http://dx.doi.org/10.1145/3010915.3010993

36. Da-Yuan Huang, Ming-Chang Tsai, Ying-Chao Tung, Min-Lun Tsai, Yen-Ting Yeh, Liwei Chan, Yi-Ping Hung, and Mike Y. Chen. 2014. TouchSense: Expanding Touchscreen Input Vocabulary Using Different Areas of Users' Finger Pads. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '14)*. ACM, New York, NY, USA, 189–192. DOI: http://dx.doi.org/10.1145/2556288.2557258

37. Jong Wook Kim and Frank E. Ritter. 2015. Learning, Forgetting, and Relearning for Keystroke- and Mouse-Driven Tasks: Relearning Is Important. *Human–Computer Interaction* 30, 1 (Jan. 2015), 1–33. DOI:http://dx.doi.org/10.1080/07370024.2013.828564

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 (NordiCHI '08)*. ACM, New York, NY, USA, 239–248. DOI:http://dx.doi.org/10.1145/1463160.1463186

39. Per-Ola Kristensson and Shumin Zhai. 2004. SHARK2: A Large Vocabulary Shorthand Writing System for Pen-based Computers. In *Proceedings of the 17th Annual ACM Symposium on User Interface Software and Technology (UIST '04)*. ACM, New York, NY, USA, 43–52. DOI:http://dx.doi.org/10.1145/1029632.1029640

40. 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 (CHI '94)*. ACM, New York, NY, USA, 258–264. DOI: http://dx.doi.org/10.1145/191666.191759

41. Gordon Kurtenbach, George W. Fitzmaurice, Russell N. Owen, and Thomas Baudel. 1999. The Hotbox: Efficient Access to a Large Number of Menu-items. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '99)*. ACM, New York, NY, USA, 231–237. DOI: http://dx.doi.org/10.1145/302979.303047

42. Gordon Kurtenbach, Thomas P. Moran, and William Buxton. 1994. Contextual animation of gestural commands. In *Computer Graphics Forum*, Vol. 13. Wiley Online Library, 305–314. DOI: http://dx.doi.org/10.1111/1467-8659.1350305

43. Gordon Paul Kurtenbach. 1993. *The design and evaluation of marking menus*. Ph.D. Dissertation. University of Toronto. https://autodeskresearch.com/pdf/theses/kurtenbach-phd.pdf

44. Gordon P. Kurtenbach, Abigail J. Sellen, and William A. S. Buxton. 1993. An Empirical Evaluation of Some Articulatory and Cognitive Aspects of Marking Menus. *Hum.-Comput. Interact.* 8, 1 (March 1993), 1–23. DOI: http://dx.doi.org/10.1207/s15327051hci0801_1

45. Benjamin Lafreniere, Carl Gutwin, and Andy Cockburn. 2017. Investigating the Post-Training Persistence of Expert Interaction Techniques. *ACM Trans. Comput.-Hum. Interact.* 24, 4 (Aug. 2017), 29:1–29:46. DOI:http://dx.doi.org/10.1145/3119928

46. 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 (CHI '16)*. ACM, New York, NY, USA, 4663–4674. DOI:http://dx.doi.org/10.1145/2858036.2858166

47. David M. Lane, H. Albert Napier, S. Camille Peres, and Aniko Sandor. 2005. Hidden Costs of Graphical User Interfaces: Failure to Make the Transition from Menus and Icon Toolbars to Keyboard Shortcuts. *International Journal of Human-Computer Interaction* 18, 2 (May 2005), 133–144. DOI: http://dx.doi.org/10.1207/s15327590ijhc1802_1

48. Wanyu Liu, Gilles Bailly, and Andrew Howes. 2017. Effects of Frequency Distribution on Linear Menu Performance. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems (CHI '17)*. ACM, New York, NY, USA, 1307–1312. DOI:http://dx.doi.org/10.1145/3025453.3025707

49. 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 (CHI '13)*. ACM, New York, NY, USA, 573–582. DOI: http://dx.doi.org/10.1145/2470654.2470735

50. Joseph Malloch, Carla F. Griggio, Joanna McGrenere, and Wendy E. Mackay. 2017. Fieldward and Pathward: Dynamic Guides for Defining Your Own Gestures. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems (CHI '17)*. ACM, New York, NY, USA, 4266–4277. DOI: http://dx.doi.org/10.1145/3025453.3025764

51. Damien Masson, Alix Goguey, Sylvain Malacria, and Géry Casiez. 2017. WhichFingers: Identifying Fingers on Touch Surfaces and Keyboards using Vibration Sensors. In *Proceedings of the 30th Annual Symposium on User Interface Software and Technology (UIST '17)*. ACM, New York, NY, USA. DOI: http://dx.doi.org/10.1145/3126594.3126619

52. Hugh McLoone, Ken Hinckley, and Edward Cutrell. 2003. Bimanual interaction on the microsoft office keyboard. In *INTERACT'03*. 49–56. https://books.google.ca/books?hl=en&lr=&id=PTg0fVYqgCcC&oi=fnd&pg=PA49&dq=Bimanualinteractiononthe+microsoft+office+keyboard.&ots=O9MMAHgzs1&sig=eHGSdZxTfU0aUf8jG0ZxCDacWSw

53. Jakob Nielsen. 1992. Finding Usability Problems Through Heuristic Evaluation. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '92)*. ACM, New York, NY, USA, 373–380. DOI:http://dx.doi.org/10.1145/142750.142834

54. Jakob Nielsen and Jonathan Levy. 1994. Measuring Usability: Preference vs. Performance. *Commun. ACM* 37, 4 (April 1994), 66–75. DOI: http://dx.doi.org/10.1145/175276.175282

55. Erik Nilsen, HeeSen Jong, Judith S. Olson, Kevin Biolsi, Henry Rueter, and Sharon Mutter. 1993. The Growth of Software Skill: A Longitudinal Look at Learning & Performance. In *Proceedings of the INTERACT '93 and CHI '93 Conference on Human Factors in Computing Systems (CHI '93)*. ACM, New York, NY, USA, 149–156. DOI: http://dx.doi.org/10.1145/169059.169126

56. Don Norman. 2013. *The design of everyday things: Revised and expanded edition*. Basic Books (AZ).

57. 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 (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

58. A. Oka and M. Hashimoto. 2013. Marker-less piano fingering recognition using sequential depth images. In *The 19th Korea-Japan Joint Workshop on Frontiers of Computer Vision*. 1–4. DOI: http://dx.doi.org/10.1109/FCV.2013.6485449

59. Halla Olafsdottir and Caroline Appert. 2014. Multi-touch Gestures for Discrete and Continuous Control. In *Proceedings of the 2014 International Working Conference on Advanced Visual Interfaces (AVI '14)*. ACM, New York, NY, USA, 177–184. DOI: http://dx.doi.org/10.1145/2598153.2598169

60. Stephen Olejnik and James Algina. 2003. Generalized eta and omega squared statistics: measures of effect size for some common research designs. *Psychological Methods* 8, 4 (Dec. 2003), 434–447. DOI: http://dx.doi.org/10.1037/1082-989X.8.4.434

61. S. Camille Peres, Michael D. Fleetwood, Minmin Yang, Franklin P. Tamborello, and Danielle Paige Smith. 2005. Pros, Cons, and Changing Behavior: An Application in the Use of the Keyboard to Issue Commands. *Proceedings of the Human Factors and Ergonomics Society Annual Meeting* 49, 5 (Sept. 2005), 637–641. DOI: http://dx.doi.org/10.1177/154193120504900501

62. Deniese Pierotti. 1995. Heuristic evaluation-a system checklist. *Xerox Corporation* (1995). http://eitidaten.fh-pforzheim.de/daten/mitarbeiter/blankenbach/vorlesungen/GUI/Heuristic_Evaluation_Checklist_stcsig_org.pdf

63. Thomas Pietrzak, Sylvain Malacria, and Gilles Bailly. 2014. CtrlMouse Et TouchCtrl: Duplicating Mode Delimiters on the Mouse. In *Proceedings of the 26th Conference on L'Interaction Homme-Machine (IHM '14)*. ACM, New York, NY, USA, 38–47. DOI: http://dx.doi.org/10.1145/2670444.2670447

64. Jun Rekimoto, Takaaki Ishizawa, Carsten Schwesig, and Haruo Oba. 2003. PreSense: Interaction Techniques for Finger Sensing Input Devices. In *Proceedings of the 16th Annual ACM Symposium on User Interface Software and Technology (UIST '03)*. ACM, New York, NY, USA, 203–212. DOI: http://dx.doi.org/10.1145/964696.964719

65. Mary Beth Rosson. 1983. Patterns of Experience in Text Editing. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '83)*. ACM, New York, NY, USA, 171–175. DOI: http://dx.doi.org/10.1145/800045.801604

66. 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 (CHI '12)*. ACM, New York, NY, USA, 257–266. DOI: http://dx.doi.org/10.1145/2207676.2207713

67. Joey Scarr, Andy Cockburn, Carl Gutwin, Andrea Bunt, and Jared E. Cechanowicz. 2014. The Usability of CommandMaps in Realistic Tasks. In *Proceedings of the 32Nd Annual ACM Conference on Human Factors in Computing Systems (CHI '14)*. ACM, New York, NY, USA, 2241–2250. DOI: http://dx.doi.org/10.1145/2556288.2556976

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 (CHI '13)*. ACM, New York, NY, USA, 3139–3148. DOI: http://dx.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 (CHI '11)*. ACM, New York, NY, USA, 2741–2750. DOI: http://dx.doi.org/10.1145/1978942.1979348

70. Richard A. Schmidt, Douglas E. Young, Stephan Swinnen, and Diane C. Shapiro. 1989. Summary knowledge of results for skill acquisition: Support for the guidance hypothesis. *Journal of Experimental Psychology: Learning, Memory, and Cognition* 15, 2 (1989), 352–359. DOI: http://dx.doi.org/10.1037/0278-7393.15.2.352

71. 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 (CHI '16)*. ACM, New York, NY, USA, 4687–4698. DOI: http://dx.doi.org/10.1145/2858036.2858412

72. Atsushi Sugiura and Yoshiyuki Koseki. 1998. A User Interface Using Fingerprint Recognition: Holding Commands and Data Objects on Fingers. In *Proceedings of the 11th Annual ACM Symposium on User Interface Software and Technology (UIST '98)*. ACM, New York, NY, USA, 71–79. DOI: http://dx.doi.org/10.1145/288392.288575

73. Hemant Bhaskar Surale, Fabrice Matulic, and Daniel Vogel. 2017. Experimental Analysis of Mode Switching Techniques in Touch-based User Interfaces. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems (CHI '17)*. ACM, New York, NY, USA, 3267–3280. DOI: http://dx.doi.org/10.1145/3025453.3025865

74. Susanne Tak, Piet Westendorp, and Iris van Rooij. 2013. Satisficing and the Use of Keyboard Shortcuts: Being Good Enough Is Enough? *Interacting with Computers* 25, 5 (Sept. 2013), 404–416. DOI: http://dx.doi.org/10.1093/iwc/iwt016

75. Md. Sami Uddin and Carl Gutwin. 2016a. Rapid Command Selection on Multi-Touch Tablets with Single-Handed HandMark Menus. In *Proceedings of the 2016 ACM International Conference on Interactive Surfaces and Spaces (ISS '16)*. ACM, New York, NY, USA, 205–214. DOI: http://dx.doi.org/10.1145/2992154.2992172

76. Md. Sami Uddin and Carl Gutwin. 2016b. Single-Handed HandMark Menus: Rapid Command Selection on Tablets. In *Proceedings of the 2016 ACM International Conference on Interactive Surfaces and Spaces (ISS '16)*. ACM, New York, NY, USA, 453–456. DOI:http://dx.doi.org/10.1145/2992154.2996871

77. Md. Sami Uddin, Carl Gutwin, and Benjamin Lafreniere. 2016. HandMark Menus: Rapid Command Selection and Large Command Sets on Multi-Touch Displays. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems (CHI '16)*. ACM, New York, NY, USA, 5836–5848. DOI: http://dx.doi.org/10.1145/2858036.2858211

78. Feng Wang, Xiang Cao, Xiangshi Ren, and Pourang Irani. 2009. Detecting and Leveraging Finger Orientation for Interaction with Direct-touch Surfaces. In *Proceedings of the 22Nd Annual ACM Symposium on User Interface Software and Technology (UIST '09)*. ACM, New York, NY, USA, 23–32. DOI: http://dx.doi.org/10.1145/1622176.1622182

79. Feng Wang and Xiangshi Ren. 2009. Empirical Evaluation for Finger Input Properties in Multi-touch Interaction. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '09)*. ACM, New York, NY, USA, 1063–1072. DOI: http://dx.doi.org/10.1145/1518701.1518864

80. Jingtao Wang and John Canny. 2004. FingerSense: Augmenting Expressiveness to Physical Pushing Button by Fingertip Identification. In *CHI '04 Extended Abstracts on Human Factors in Computing Systems (CHI EA '04)*. ACM, New York, NY, USA, 1267–1270. DOI:http://dx.doi.org/10.1145/985921.986040

81. Ian H. Witten, John G. Cleary, and Saul Greenberg. 1984. On frequency-based menu-splitting algorithms. *International Journal of Man-Machine Studies* 21, 2 (Aug. 1984), 135–148. DOI: http://dx.doi.org/10.1016/S0020-7373(84)80063-2

82. Shumin Zhai and Per-Ola Kristensson. 2003. Shorthand Writing on Stylus Keyboard. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '03)*. ACM, New York, NY, USA, 97–104. DOI:http://dx.doi.org/10.1145/642611.642630

83. Shumin Zhai and Per Ola Kristensson. 2012. The Word-gesture Keyboard: Reimagining Keyboard Interaction. *Commun. ACM* 55, 9 (Sept. 2012), 91–101. DOI:http://dx.doi.org/10.1145/2330667.2330689

84. Shumin Zhai, Per Ola Kristensson, Pengjun Gong, Michael Greiner, Shilei Allen Peng, Liang Mico Liu, and Anthony Dunnigan. 2009. Shapewriter on the Iphone: From the Laboratory to the Real World. In *CHI '09 Extended Abstracts on Human Factors in Computing Systems (CHI EA '09)*. 2667–2670.

85. Haimo Zhang and Yang Li. 2014. GestKeyboard: Enabling Gesture-based Interaction on Ordinary Physical Keyboard. In *Proceedings of the 32Nd Annual ACM Conference on Human Factors in Computing Systems (CHI '14)*. ACM, New York, NY, USA, 1675–1684. DOI: http://dx.doi.org/10.1145/2556288.2557362

86. 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 (CHI '18)*. ACM, New York, NY, USA, 249:1–249:14. DOI: http://dx.doi.org/10.1145/3173574.3173823

87. Jingjie Zheng and Daniel Vogel. 2016. Finger-Aware Shortcuts. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems (CHI '16)*. ACM, New York, NY, USA, 4274–4285. DOI: http://dx.doi.org/10.1145/2858036.2858355

APPENDIX

This appendix provides the questions asked during the semi-structured interview for *Study 1: Interview with Expert Users*.

Demographics

- What is your age and gender?
- What is your occupation? If you are a student, what is your major and what is your level of study?
- What is your dominant hand?

Computer Usage

- How many hours per week do you use laptop/desktop computers?
- What operating systems do you use most often?
- What do you often do with computers?
- What tasks do you typically perform?
- What applications do you use most often? Provide at least 10 applications sorted by frequency of use.

Shortcut Use in a Specific Application

- What is your favourite application?
- In this application, how often do you use keyboard shortcuts?
- In this application, please recall as many keyboard shortcuts as you can without referring to that application or any related material.
- In this application, what keyboard shortcuts are those that you frequently use but are uncertain whether they are correct or need to be frequently looked up?
- In this application, what actions do you frequently perform, but could never memorise the keyboard shortcut for an extended period of time?
- In this application, what keyboard shortcuts did you expect to work in one way, but in reality a different action is triggered?

Shortcut Use in General

- When do you actively learn a new keyboard shortcut?
- How do you find out what a keyboard shortcut is mapped to a command?
- How do you learn new keyboard shortcuts? What strategies do you often apply?
- How often do you have to rehearse a learned keyboard shortcut?
- What kinds of keyboard shortcuts do you tend to need to rehearse?
- What kinds of keyboard shortcuts do you tend to remember for a very long time?
- What kinds of keyboard shortcuts do you tend to forget or get confused?
- When articulating keyboard shortcuts, do you need to think about what keys to press?

Ratings

Appendix Note: Ratings used a five point scale with respect to the question where 1 was "worst/rarely/difficult" and 5 was "best/often/easy". The scale used for each question is shown with range descriptions (e.g. "[1 difficult ... 5 easy]").

- In general, how do you find learning keyboard shortcuts?
  [1 difficult ... 5 easy]
  Why did you give the above rating?
- In general, how do you find executing keyboard shortcuts?
  [1 difficult ... 5 easy]
  Why did you give the above rating?
- How often do you make a physical error while executing keyboard shortcuts?
  [1 rarely ... 5 often]
  Why did you give the above rating?
- In general, how do you find knowing whether a keyboard shortcut has successfully activated the desired command?
  [1 difficult ... 5 easy]
  Why did you give the above rating?

Please rate the following aspects of keyboard shortcuts:

- Learning [1 worst ... 5 best] Why?
- Ease of use [1 worst ... 5 best] Why?
- Accuracy [1 worst ... 5 best] Why?
- Speed [1 worst ... 5 best] Why?
- Fatigue [1 worst ... 5 best] Why?

Concluding Thoughts

- What do you think are the key issues of keyboard shortcuts? How would you address these issues?
- Do you have other comments?