Understanding and Exploiting Spatial Memory in the Design of Efficient Command Selection Interfaces

---

A thesis
submitted in partial fulfilment
of the requirements for the Degree
of
Doctor of Philosophy
in the
University of Canterbury
by
Joseph Scarr

Supervision and Examination Committee

Professor Andy Cockburn	Supervisor
Professor Tim Bell	Co-supervisor
TBC	External Examiner
TBC	External Examiner

Department of Computer Science and Software Engineering
University of Canterbury
2014

Abstract

Humans have a strong natural ability to remember item locations. In graphical user interfaces, this ability is one of the primary mechanisms by which users become efficient. However, there are two ways in which modern applications often fail to exploit the potential of spatial memory. First, they overuse hierarchical structures such as cascading menus, which slows down interaction for expert users who already know item locations; and second, they move items around, most commonly in response to changing display geometry. The three goals of this thesis are therefore to (1) develop a better understanding of human spatial memory in the context of user interfaces; (2) design and validate efficient command-selection interfaces based on the strength of spatial memory; and (3) design and validate interface strategies that allow users to maintain spatial memory even when display geometry changes.

Addressing goal (1), a comprehensive literature review of spatial memory for user interfaces is presented. The review covers underlying psychological models of spatial memory, the observable properties of spatial memory, and existing applications of spatial memory to human-computer interaction. In addition to informing the research in this thesis, the review is intended to provide a useful summary of the state of spatial memory research for scientists in HCI, as well as providing a set of design guidelines on spatial memory for practitioners.

Addressing goal (2), this thesis presents the design and evaluation of two related user interface techniques, CommandMaps and StencilMaps. The CommandMap is a spatially stable interface with a flattened hierarchy, intended as a replacement for cascading menu systems. Theoretical performance predictions indicate that CommandMaps should be significantly faster than traditional user interfaces such as menus and the Microsoft Office Ribbon, and laboratory-based empirical studies of command selection confirm these predictions. These positive results motivated the design and implementation of two real-world CommandMap user interfaces based on Microsoft Word and Pinta (an open-source image editing application). Evaluation results confirmed that CommandMaps continue

to demonstrate performance and subjective advantages in the context of actual tasks, including interleaved command selection, typing, and direct manipulation. Qualitative data gathered from interviews, questionnaires, and conversations provide substantial insight into users' reactions to CommandMaps, leading to a set of design recommendations regarding when and how they should be implemented in real applications.

One design limitation identified during CommandMap evaluations was that novice users could be initially overwhelmed by the number of controls displayed at once. To address this concern, an extension to the CommandMap, called a StencilMap, was designed and evaluated. By using a stencil overlay to de-emphasise more advanced controls, the StencilMap directs users' visual search to a subset of controls they are most likely to need. Then, when novice users progress to the full interface, they can utilise their existing knowledge of command locations. An initial study shows that stencils are more effective at guiding visual search than ephemeral adaptation, another subset emphasis technique; however, users' spatial learning decreases as the amount of guidance increases. A second study compared StencilMaps to a palette-based subset interface, which displays the most likely commands in a ready-to-hand tool panel. Results show that StencilMaps enable stronger learning of the full UI compared to the palette approach.

Addressing goal (3), this thesis presents an investigation of how interfaces can be adapted to changing interface constraints while still supporting the user's memory for item locations. A human factors study on *spatially consistent* transformations was conducted, with results showing that people's spatial memory is only minimally disrupted by geometric transformations (such as scaling, translation, or perspective distortion), as long as the set of items in a display is transformed as a whole. This idea is then applied to a file browser layout: by scaling the item grid when the parent window is resized, rather than reflowing items, memory for item locations can be maintained. A second study validates this idea, showing that a scaling interface outperforms both reflow and scrolling-based techniques for revisitation when windows are resized.

In summary, the contributions of this thesis are: (1) an in-depth literature review of spatial memory in psychology and HCI, which is intended to inform designers and future researchers as well as the material in this thesis; (2) the design, implementation and evaluation of a new interface, the CommandMap, which

shows that spatial stability and hierarchy flattening enable a high ceiling of expert performance; (3) the design of a stencil overlay technique to help novice users find commands, and an evaluation highlighting the key trade-off between helping users and allowing them to learn; and (4) empirical evidence showing that most types of whole-interface transformations have a small effect on spatial memory, and that correspondingly, scaling interfaces outperform reflowing interfaces under changing window constraints.

Publications Arising from this Thesis

Much of the work contained in this thesis has been published in peer-reviewed journals and conferences, listed below. The corresponding chapters on which each publication is based are noted in parentheses.

1. **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. **[CHI 2012 Best Paper award.]** (Chapter 5).

2. **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. **[CHI 2013 Honorable Mention award.] [CHI 2013 RepliCHI award.]** (Chapter 8).

3. **Joey Scarr**, Andy Cockburn, and Carl Gutwin. Supporting and exploiting spatial memory in user interfaces. 2013. *Foundations and Trends® in Human-Computer Interaction* 6, 1, 1-84. (Chapters 1, 2, 3, 4).

4. **Joey Scarr**, Andy Cockburn, Carl Gutwin, Andrea Bunt, and Jared Cechanowicz. 2014. The usability of CommandMaps in realistic tasks. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (CHI '14). To appear. (Chapter 6).

5. **Joey Scarr**. Exploiting spatial memory to design efficient command interfaces. In *Extended Abstracts on Human Factors in Computing Systems* (CHI EA '13). ACM, New York, NY, USA, 1961-1964. (Chapters 5, 7, 8).

In addition, I contributed to the following publications during my doctoral studies. Each of these papers is related either to spatial memory or the broader

issue of transitioning to expert behaviour with user interfaces. The papers mentioned below are included as appendices.

6. 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* (UIST '13). ACM, New York, NY, USA, 321-330. (Appendix A).

7. Carl Gutwin, Andy Cockburn, **Joey Scarr**, Sylvain Malacria, Scott Olson. 2014. Faster command selection on tablets with FastTap. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (CHI '14). To appear. (Appendix B).

8. Andy Cockburn, Carl Gutwin, **Joey Scarr**, Sylvain Malacria. Supporting novice to expert transitions in user interfaces. *ACM Computing Surveys*. In submission. (Appendix C).

Declaration and Technical Acknowledgements

With some exceptions, the work contained within this thesis is my own. My supervisor, Professor Andy Cockburn, and Professor Carl Gutwin from the University of Saskatchewan both provided ideas, inspiration, and feedback throughout the duration of my research, and their input is incorporated throughout this document. My other co-authors, Sylvain Malacria and Andrea Bunt, also contributed to the publications on which parts of this thesis is based, and accordingly some of their ideas and suggestions for improvement are also included in this document.

The experiments performed in Chapters 6, 7, and 8 were partially, and in some cases completely, conducted by Jared Cechanowicz in Professor Carl Gutwin's lab at the University of Saskatchewan. However, the experimental design, implementation, analysis and writeup in each case was done by me. Additionally, the CommandMap implementation for Pinta (Chapter 6) was initially developed by Scott Olson, also from the University of Saskatchewan.

Many thanks go to my research collaborators who helped bring this thesis to fruition.

Ethical Considerations

The experiments described in this thesis were conducted in two locations: the University of Canterbury and the University of Saskatchewan. Experiments in Chapters 6, 7 and 8 were conducted at the University of Saskatchewan under Carl Gutwin's supervision, and were covered by his ethics approval. All other experiments were approved by the University of Canterbury's Human Ethics Committee (number HEC2012/06).

The privacy, comfort and well-being of all participants was carefully considered during this research, and all participants maintained the right to withdraw and have their data destroyed at any point. Participants in experiments that were conducted at the University of Canterbury completed written consent forms; an example can be found in Appendix D.

Acknowledgements

I would like to thank the following people, each of whom made substantial contributions to the thesis-writing period of my life. Without them, this thesis would have been significantly less enjoyable to create, and most likely would not exist at all.

First and foremost, thank you to my supervisor, Professor Andy Cockburn, for your wisdom, guidance, encouragement, and friendship throughout the research process. Your endless enthusiasm for our various research ideas provided a healthy balance to my skepticism, and I most definitely could not have completed this project without you. Thanks also to Professor Tim Bell, my co-supervisor, for putting your name on the progress reports and providing moral support.

Thanks to Professor Carl Gutwin of the University of Saskatchewan, for contributing ideas, words, and generally being involved in many parts of my research; and thank you for being such a gracious host during my visit to Saskatoon. Thanks also to the faculty and students at the University of British Columbia and the University of Calgary for hosting me and exchanging research ideas during my visits in 2012.

Thank you to Sweet Yee, for supporting me during my research, for putting up with my absence for long periods of time, for making my colleagues jealous with your home-made lunches, and finally for marrying me. Coming home to you was always the highlight of my day.

Thanks to my parents, Karen and Larry, for introducing me to computers at a young age, and for nurturing my curiosity and independence. Thanks to Nik, Edward, Lukas, Tuan, and Glen for coming tramping with me regularly and making sure I saw some sunlight. Thanks to Chris, Ange, Jonny and James for being generally excellent friends and providing some much-needed non-computer talk.

Thank you to my HCI lab colleagues (Sylvain, Mathieu, Stephen, Philip, and Joshua) for your friendship and for many interesting discussions. Extra thanks to Sylvain and Mathieu for participating in literally hundreds of lunchtime games of 7 Wonders.

Finally, thank you to all my study participants for donating your valuable time. Without your data, my conclusions would have been substantially more difficult to justify.

This thesis was supported by the *Brownlie Scholarship*, awarded to the top applicant for a University of Canterbury Doctoral Scholarship each year. Additional funding was provided by the *New Zealand Royal Society Marsden Grant 10-UOC-020*. Financial support for travel between labs in Canada was provided by the *Natural Sciences and Engineering Research Council of Canada* and the *GRAND NCE*.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

Spatial memory plays an important part in our day-to-day lives. In the physical world, our ability to recall spatial information enables us to locate items in our homes, semi-autonomously navigate in previously-encountered environments, drive to work without the use of a map, and perform many other activities that would otherwise require cognitive and physical effort.

Evidence that spatial memory is a particularly powerful capability of the human brain can be found in mnemonic literature [212, 16], and dates back thousands of years. The ancient Greeks and Romans used spatial mental organisations based on the architecture of the time, known as *memory palaces*, to connect, organise, and memorise unfamiliar ideas, particularly for the purposes of public speaking. This was called *the method of loci*. By embedding key images representing topics in a mental representation of a spatial environment, such as the rooms in a familiar building, orators were able to memorise extremely long sequences of topics. These could then be retrieved by mentally walking through the building and viewing the images in their respective spatial locations [212, 16].

In human-computer interaction, spatial memory provides many of the same benefits as in the real world: a strong spatial knowledge of interface layouts and control locations, particularly in graphical user interfaces, allows users to substantially reduce the cognitive and physical effort required for interaction. Evidence for the benefits provided by spatial memory can be found in the strong correlation between measures of spatial ability and interface performance [66, 125, 156]. Users who are unfamiliar with an interface must spend considerable time searching for controls, because the time to perform visual search is proportional to the number of items [93, 155, 176, 33]. In contrast, users who are familiar with a spatially-stable interface do not need to carry out visual search, and can instead simply retrieve item locations from memory. This is much faster than searching

because retrieval time is a logarithmic function of the number of items [86, 96, 33].

Furthermore, extensive spatial knowledge of an application's controls enables interaction *automaticity* [174], which substantially frees the user's cognitive resources from the need to consider interface mechanisms, allowing the user to instead focus on higher-level task considerations. Spatial knowledge of the locations of controls can also decrease the frustration that arises from the need to search for unfamiliar controls or controls that have moved. This suggests that allowing and encouraging users to utilise their spatial memory whenever possible should be an important goal for interface designers.

In current user interfaces, however, this goal is often ignored in favour of other considerations. In particular, hierarchical interfaces such as cascading menus unnecessarily hinder users who are familiar with item locations: even though a user might know exactly where an item is, they are forced to navigate a hierarchy in order to find it. In a similar vein, spatial memory is disrupted when on-screen items are moved: for example, interfaces with changeable geometry (such as desktop windows) or variable content (such as file browsers) often re-arrange items when the geometry changes or when items are added or removed from the display.

This thesis aims to better understand spatial memory and its importance in HCI, as well as provide solutions to the under-utilisation and disruption of spatial memory. To this end, a comprehensive literature review of spatial memory as it relates to user interfaces is presented. This review inspires the design, implementation and evaluation of the CommandMap, a hierarchically flat interface that exploits spatial memory to enable rapid item selection. Next, the design and evaluation of the StencilMap, an extension to the CommandMap intended to improve novice visual search, is presented. Finally, this thesis investigates ways to keep item layouts spatially consistent when display geometries change, and develops fundamental understanding of the impact of various forms of spatially consistent transformation.

The following subsections formally define the research goals of this thesis and the approach taken to achieve them, identify the specific research contributions of the thesis, and outline the thesis structure.

1.1 Research Goals

The overarching goals of this thesis are to better understand spatial memory and exploit it to design more efficient user interfaces. These goals are expanded into three concrete subgoals, described below; criteria for judging the degree to which these subgoals are achieved are also provided.

1. *Understand the capabilities and characteristics of spatial memory in the context of user interfaces.* Work towards this goal will result in a review of the models and theories underlying spatial memory, the empirical characteristics of spatial memory when interacting with computers (and in related contexts), and the documented ways in which existing commercial and research UIs affect, and are affected by, spatial memory.

2. *Exploit the capabilities of spatial memory to design, implement, and evaluate efficient command-selection interfaces.* This goal will result in the design and implementation of new command-selection techniques that empirically and subjectively outperform standard techniques through the exploitation of spatial memory.

3. *Design and evaluate new interface techniques that maintain spatial memory in situations when it would normally be disrupted.* This goal will result in new, spatially-consistent design solutions to situations where items must move to accommodate changes in display constraints.

1.2 Research Contributions

This thesis makes five primary contributions to spatial memory research in HCI. These are:

1. A comprehensive review of spatial memory in HCI, covering the psychology of spatial memory, the properties of spatial memory, and the interaction between spatial memory and existing user interfaces. At the time of writing, no such summary exists; this review therefore not only informs the present research, but also provides a tool for other HCI researchers to quickly become familiar with existing spatial memory literature.

2. The design and evaluation of the CommandMap, a UI technique that uses a flattened command hierarchy to enable rapid spatial retrieval of controls. Theoretical performance modelling indicates that expert users should perform better with CommandMaps than with standard hierarchical UIs. Three studies using a CommandMap mockup demonstrate that experts can select commands more quickly with CommandMaps than with menus and the Microsoft Office Ribbon, that novice CommandMap performance is similar to that of standard techniques, and that pop-up CommandMaps that extend past the window border perform better than CommandMaps that scale to the window bounds. The methodology used in these studies, however, is designed to maximise statistical power at the cost of external validity, leaving some questions about real-world use unanswered.

3. The real-world implementation of CommandMaps and evaluation of their use in realistic tasks. In order to answer questions about realistic CommandMap use, CommandMap interfaces were implemented for both Microsoft Word and Pinta (an open-source image editing application). Two studies on these real-world CommandMap implementations provide deeper insights into users' opinions of CommandMaps, and demonstrate that the advantages of CommandMaps are largely maintained in realistic tasks. The studies also reveal some potential issues with the initial perceptions and performance of novice users.

4. The design and evaluation of the StencilMap, a training interface based on the CommandMap intended to assist novice users. The StencilMap uses a stencil overlay to highlight a subset of the most relevant commands. An initial experiment shows that StencilMaps improve visual search times over standard CommandMaps and ephemeral highlighting [57, 123], but also indicates that item locations are learnt less well when greater visual assistance is provided. A second experiment compares StencilMaps to a spatially stable palette, showing that StencilMaps allowed users to better learn item locations, at the cost of a small initial time penalty. The results of both experiments reveal several issues in regard to evaluating the efficacy of subset interfaces.

5. An investigation of the effects of spatially-consistent transformations on spatial memory. CommandMaps, StencilMaps, and other spatially arranged interfaces (such as file browsers) require spatial stability to maintain user performance. However, when window or display dimensions change, interfaces often abandon spatial stability – for example, by eliding and compacting groups of commands, or reflowing grid items. An experiment examining *spatially consistent transformations* (such as translation, rotation, or scaling), where item locations are kept constant relative to some frame of reference, shows that spatially-consistent transformations other than rotation have minimal effects on spatial memory, while the selection time penalty incurred by rotation is proportional to the degree of rotation. A second experiment applies this result to the resizing of file browser windows, showing that a scaling interface preserves spatial memory better in resizing windows than the common strategy of reflowing items, allowing users to maintain a high level of performance.

1.3 Structure of the Thesis

The thesis is divided into four parts: a review of spatial memory literature from psychology and HCI; the development of efficient user interfaces that exploit spatial memory; an investigation of how to maintain spatial consistency when interface constraints change; and an in-depth discussion of the work contained in this thesis, including identification of limitations and future research directions. The first three parts of the thesis are intended to address, in turn, the three research goals described in Section 1.1.

Part I, addressing Research Goal 1, reviews spatial memory in the context of HCI. Chapter 2 summarises relevant psychological theories and models of spatial memory, and differentiates memory for object locations on screens or tabletops from navigation memory. Chapter 3 looks at the empirically observable properties of spatial memory, such as retrieval time, capacity, and duration. Chapter 4 studies spatial memory in HCI, including existing interfaces that exploit and disrupt spatial memory, and studies of how spatial memory is affected in variable-view (such as pan-and-zoom) interfaces.

Part II, addressing Research Goal 2, is focused on the design and evaluation

of command-selection interfaces that exploit spatial memory. The design of the CommandMap is presented in Chapter 5, and is shown to allow faster command selection than menus and the Microsoft Office Ribbon. Chapter 6 details two real-world implementations of CommandMaps, as well as two studies examining their use in realistic tasks. The results provide greater insight into user opinions of CommandMaps, including when they are most advantageous. Chapter 7 presents the StencilMap design, which investigates the use of stencils to help novice users find item locations. The results of two studies show that StencilMaps assist visual search better than other item-highlighting techniques, and that they allow users to transition to the full, "expert" interface without having to re-learn item locations.

Part III addresses Research Goal 3: providing new solutions for situations where spatial memory is disrupted. Chapter 8 focuses on the problem of moving items around when window constraints change. An initial experiment studies the robustness of spatial memory when a spatially consistent interface undergoes a geometric transformation (such as scaling). The results show that all transformation types, apart from rotation, allow users to maintain a level of item selection performance close to their level before the transformation occurred. The findings are then applied to file browser interfaces, with a second experiment showing that scaling an item grid allows for faster item selection when the window size changes than reflowing items from left-to-right.

Finally, Part IV provides a summary and discussion of the thesis, as well as directions for future work.

Part I

Understanding Spatial Memory

Chapter 2

The Psychology of Spatial Memory

This chapter provides a high-level introduction to relevant psychological models and theories of spatial memory. Since this thesis concerns the application of spatial memory to HCI, it is tempting to take a black-box approach and limit analysis to the empirically observable characteristics that occur during interaction; however, as demonstrated in Chapter 3, some knowledge of the underlying psychological theory is necessary to better comprehend the observed phenomena. It is worth noting that although an attempt is made to draw a distinction between models in this chapter and empirical work in Chapter 3, there is inevitably some overlap between the two, as empirical work often informs the development of models and vice versa.

First, this chapter introduces and justifies a separation between the concerns of spatial memory in navigation and spatial memory for object locations (as pertinent to computer displays). Spatial memory in navigation is only partially correlated with object location memory, and since this thesis concerns memory for items in mostly static displays (i.e., computer interfaces), this review focuses primarily on the latter. Second, it summarises the literature on short-term memory, focusing on Baddeley's working memory model [11]. Third, it examines models of long-term memory (LTM), including theories of how short-term memories are consolidated into LTM, theories of forgetting, models of skill acquisition, and the amount of effort involved in learning. Finally, it examines ways in which spatial information is encoded relative to its surroundings, discussing reference frames and landmarks.

This chapter focus on theories relevant to *spatial* memory; for an overview of memory in general, the reader should refer to Baddeley [9].

2.1 Navigation vs. memory for object locations

A distinction is made in this work between two classes of spatial tasks: navigation, and remembering the locations of objects in a static display. Maguire et al. [135] noted the general consensus in the field of neuroscience that "navigation is not the same as table-top tests of spatial memory [...] and that direct inferences cannot be made about one from the other" (p. 171). Indeed, the two task types, while both "spatial", are quite different: when remembering object locations on a table-top or display, the entire space is generally static and visible; conversely, the user's viewpoint during navigation tasks is dynamic, with only part of the space visible at any one time. Hegarty et al. [83] showed that ability in the two types of spatial tasks is only partially correlated. Furthermore, memory-impaired patients exist who have difficulty with navigation, but not table-top memory [139, 76], and vice versa [134].

Siegel and White's seminal framework [182] of the development of spatial knowledge in large-scale environments is one area where navigation and object memory overlap. The framework suggests that people's mental models of their environments are made up of landmark, route, and survey knowledge, where survey knowledge implies the ability to visualise a map-like overview of the space (similar to viewing a 2D arrangement of objects).

In HCI, navigation itself is relevant in the specific context of wayfinding in virtual environments (e.g., [40]), but the primary use for spatial memory in standard desktop and mobile interfaces is to remember the locations of on-screen controls. Therefore, while some of the literature discussed in this review is necessarily from the navigation domain, the focus is on memory for object locations (particularly in small-scale displays such as computer screens).

2.2 Working memory

Baddeley and Hitch's famous model of working memory [11, 7] proposes four separate subcomponents. The first is a *central executive*, which controls and runs in parallel with three slave systems: the *phonological loop*, the *visuo-spatial sketchpad*, and the more recently proposed *episodic buffer*. Since this thesis concerns spatial memory, only the visuo-spatial sketchpad is investigated here.

As suggested by the name, the purpose of the visuo-spatial sketchpad is handling visual and spatial tasks: for example, maintaining visual images (such as faces) for short periods of time, visualizing a route to a destination, or mentally manipulating objects. In situations where there are multiple ways to complete a mental task, the sketchpad can offer significant advantages: for example, there is evidence that people can remember a longer list of directions if they encode the information visuo-spatially (i.e., by visualising the route), rather than verbally remembering the directions [17]. However, maintaining this kind of information in working memory is prone to disruption by the presence of visual distractors, even when that information is not deliberately attended to [130]. As one might expect, visual and spatial stimuli are the primary disruptors of the sketchpad, rather than other sensory input [131].

While the visuo-spatial sketchpad model incorporates both spatial and visual working memory, there is evidence to suggest that the two may, in fact, be separate systems [75, 162, 171]. Tests that have been developed to measure memory span also show that the two components may be separable. In Corsi's block test [144], an array of nine blocks is placed on a table; the experimenter indicates a sequence of blocks by tapping on them, and the subject has to recreate the sequence from memory (Figure 2.1). The Corsi block test therefore utilises both sequential and spatial aspects of short-term memory, but does not require memorisation of a visual image. The corresponding test for visual memory is the visual patterns test [171], in which the subject is given a square grid of 50% randomly filled cells and has to remember which cells were filled (Figure 2.1). Concurrent visual interference decreases performance on the visual patterns test, but not on the Corsi test, while spatial interference has the converse effect, suggesting that the two tasks utilise separate cognitive subsystems [171].

Spatial information is not purely visual, since hearing and proprioception also provide us with data about the spatial relationships around us. However, since computer interfaces are primarily (though not exclusively) visual, this chapter primarily considers visual inputs to spatial memory. Attempting to distinguish between visual and spatial components of the sketchpad is therefore unnecessary for the purposes of this chapter; the effects of auditory and proprioceptive feedback are discussed later in Section 4.4.

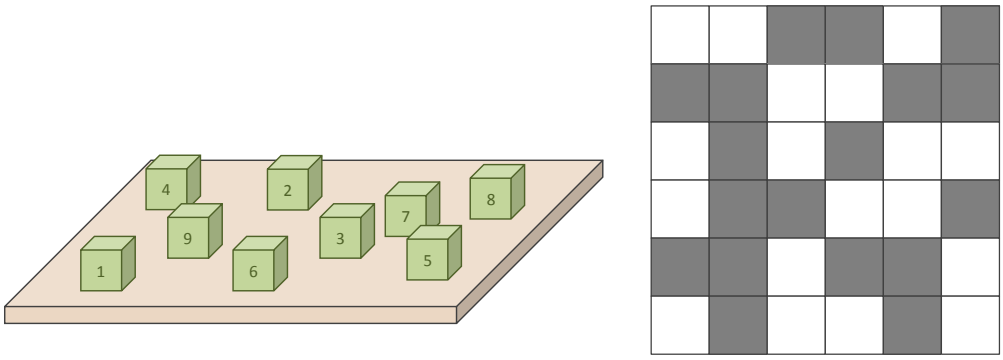


Figure 2.1: Left: In the Corsi block test, a set of blocks are laid out on a table and the experimenter taps some of them in sequence. The subject then attempts to recreate the sequence from memory. Right: The visual patterns test involves subjects being shown a square matrix where 50% of the cells are filled. The matrix is then hidden, and subjects are asked to recreate it from memory. Images adapted from [8].

2.3 Long-term memory

Long-term memory encodes associations between items in a persistent way, providing access to information over a much longer time period than what is possible with working memory. In addition to the duration of memory, long-term memory has a very large capacity, and these two characteristics make long-term memory the basis for much of human learning [128].

Long-term memory may be divided into different components [10], grouped into the two main categories of declarative and implicit memory. Declarative memory includes storage of semantic facts (e.g., "Paris is the capital of France"), and storage of episodes (i.e., episodic memory, in which people store specific details of past events). Implicit memory includes procedural skills (e.g., riding a bicycle), primed memories, and classical conditioning. Spatial memory does not generally appear as a separate component in models of long-term memory; rather, several components can play roles in spatial remembering. For example, people can remember locations as facts (e.g., "my pen was on the left side of my desk"), as physical procedures (e.g., building up 'muscle memory' for reaching for a car's gear lever), and even episodes (e.g., remembering an object's location by visualizing its most recent use).

This section investigates three separate topics that are important for long-term spatial memory. First, literature on the process of encoding and storing things in long-term memory (called consolidation) is studied, as well as the process of recall and the problem of forgetting. Second, this section considers material from the domain of cognitive skill development, which studies the progression of skill performance with practice. Finally, psychology theories of automatic and effortful learning in spatial memory are examined.

2.3.1 Storage and retrieval of long-term memories: consolidation and forgetting

A great deal of research has been carried out on how long-term memories are formed, and only a brief overview is provided here. There are several ways in which this process can occur, leading to different recall rates and levels. First, the number of times (or amount of time) an idea appears in working memory has an effect on long-term storage: the more frequently an association is used, the better it will be remembered. Second, the levels-of-processing effect [38] suggests that recall is a function of the depth of mental processing, with surface-level characteristics (e.g., words or word sounds) leading to more rapid decay than semantic processing (e.g., associations based on meaning, experience, or emotion). Third, the context in which a memory was stored becomes part of the association, leading to a phenomenon in which memories are better retrieved when a person is in the same physical and emotional context as they were when storing the memory. Fourth, the intentionality and effort put into remembering appears to affect the storage of the memory (see further discussion of effort below).

Much of our knowledge about how retrieval from long-term memory works is based on the opposite of retrieval – that is, forgetting. As with storage, there are several ways in which a memory could fail to be retrieved: for example, the association could simply decay over time (less-used associations are harder to retrieve), or the trigger for a long-term memory could be lost or confused with another (e.g., through interference).

Throughout the 20th century, the relative contribution of these mechanisms of forgetting were a source of debate amongst psychologists. Evidence was produced in favour of both simple decay over time [47] and interference from other memories [204] as factors contributing to the degradation of memory, with no

unified model fully explaining all reported phenomena. Elmes's experiment [50] is one of the few empirical studies on forgetting that focuses primarily on human spatial memory. He found that spatial memory is affected by both *proactive* interference, where the presence of existing memories makes it more difficult to form new memories (for example, re-learning the layout of an interface after a software update that causes it to change); and *retroactive* interference, where the acquisition of new memories degrades the quality of previously acquired memories (for example, learning a new interface may degrade spatial memory of previously learned interfaces).

Wixted [209] provides a comprehensive review of forgetting research in general, in which he integrates existing empirical evidence into a model primarily based on the idea of consolidation. In Wixted's model, memories take time to be consolidated into long-term memory, and mental exertion involving unrelated information interferes with that process (an example of retroactive interference). According to Wixted, acquiring new memories partially degrades the quality of memories that have been recently formed – the more recent the memory, the more affected it is by interference.

However, in the context of *spatial* recall, Tlauka et al. [198] found no effect of increased mental exertion on recall accuracy, suggesting that Wixted's model is still incomplete (or perhaps that spatial memory is different to other types of memory, an assertion supported by Elmes [50]). Despite the absence of a formal model for spatial memory, there is useful evidence to suggest that forgetting is affected by several factors – including interference (both proactive and retroactive) [50, 204], the similarity of interfering material [114], and time-based decay [47].

2.3.2 Skill acquisition

While Ebbinghaus's forgetting curve [47] accurately describes the decay of memory for declarative knowledge over time, an interesting exception can be found in the literature on skill acquisition. Baddeley [9] summarises the results of studies that show that while complex skills such as cardiac resuscitation are prone to forgetting in the same way as declarative knowledge, certain types of closed-loop skills are not: riding a bicycle or piloting an aircraft being two examples [9]. In the context of user interfaces, it is unclear to what extent spatial memory of con-

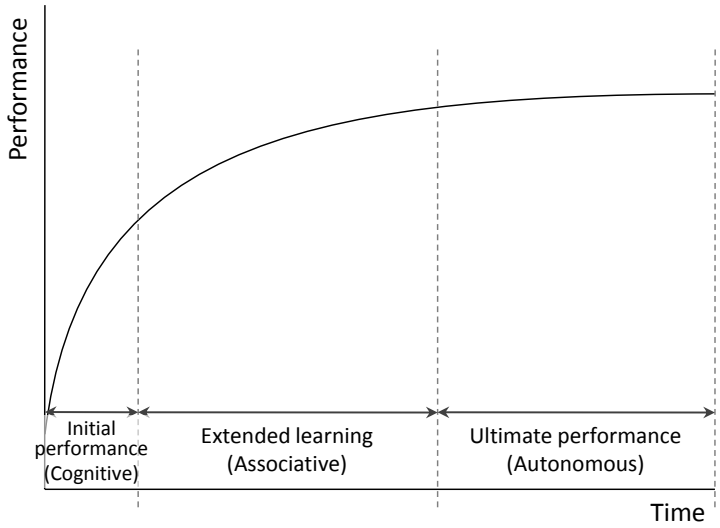


Figure 2.2: A characterisation of the progression from novice to expert skill level in a user interface. Adapted from [173].

trol locations constitutes declarative knowledge, and to what extent it is a result of muscle memory (though see Section 3.4.2 for evidence that the rate of long-term spatial forgetting is low).

Fitts and Posner's [60] influential model identifies cognitive, associative, and autonomous phases of psychomotor skill acquisition, where the cognitive phase involves consciously forming declarative mental models of the activity, the associative phase encodes motor actions in long-term memory through repetition, and the final autonomous phase allows people to perform the skill without conscious effort, freeing up cognitive resources for other tasks. Empirically speaking, however, Newell and Rosenbloom [152] showed that performance improvements followed a continuous power law rather than a set of phases; replication of this effect in HCI was provided by Card [21], who studied a single user's progression to automaticity over thousands of trials. Cockburn et al. [33] and Scarr et al. [173] used this observation to model the transition from novice to expert behaviour in user interfaces (Figure 2.2).

2.3.3 Automatic and effortful learning

In 1979, Hasher and Zacks [81] proposed a framework separating automatic and effortful memory processes. By their definition, an effortful memory process is one which requires intentional memorisation and can therefore be disrupted by concurrent task load, while an automatic process happens incidentally. They hypothesised that spatial knowledge was encoded by a largely automatic process, based on earlier results showing that spatial location memory showed little advantage of intentional over incidental learning [137, 206]. An experiment by Andrade and Meudell [6] found further evidence to support this hypothesis, with their results showing no effect of concurrent task load (counting aloud by ones, and counting aloud by sevens) on the accuracy of spatial memory for words displayed in random corners of a computer screen. Similarly, Rothkopf [170] found that subjects developed incidental memory for the on-page location of sentences after reading a 3000-word text, even though they had not been instructed to memorise locations.

However, the opposite result was found by Naveh-Benjamin [149, 150], who showed that performance in both spatial recall and recognition tasks was improved when participants had a lower concurrent task load, and when they explicitly intended to learn the information; in addition, performance was also significantly affected by individual differences.

Evidence in the navigation literature suggests that certain kinds of spatial information and location cues are more readily encoded automatically than others. Van Asselen et al. [205] had study participants walk a route through an unfamiliar building, with half focusing on the route and half not. The ability to recognise landmarks and place them in the correct order did not differ between the two groups, but those who focused on the route were able to construct better maps and made fewer mistakes when finding their way back to the beginning of the route. Self-directed exploration can also provide benefits: Hazen [82] found that children who explored a playhouse developed a more accurate knowledge of spatial layout than those who were led around by their parents. However, it is unclear whether the benefits of self-direction are due to increased attention, or deeper encodings caused by the ability to make decisions and learn from mistakes [27].

The structure in which spatial information is presented may also have an ef-

fect on whether it can be encoded automatically or not. Doeller and Burgess [44] found that object locations were learned automatically in relation to local boundaries (such as the edge of a table or frame of a computer screen), but that spatial relationships between landmarks had to be learned intentionally. There is also evidence to suggest that proactive interference makes spatial learning more difficult – in other words, the more information one attempts to learn at once, the more difficult learning becomes [41].

In the static object layouts used by computer interfaces, it is likely that spatial memory can be encoded both automatically and effortfully: while there is evidence that it largely develops as a byproduct of interaction [107, 129, 48], intentional practice has been shown in some cases to provide better long-term retention [127, 48, 27]. The interested reader should refer to Chrastil and Warren's [27] detailed review of intentional and automatic spatial learning for more information.

2.4 Spatial reference systems

Information about the location of an object cannot exist in isolation – it must be described in relation to something else. Even 'absolute' location descriptors such as Cartesian co-ordinates are defined in terms of an origin point and a set of reference axes. Human spatial memory is no different. This section discusses frames of reference and landmarks, both of which provide spatial structure in relation to which other locations can be described.

2.4.1 Frames of reference

Psychology research shows that when we learn the locations and spatial structures of objects in our environment, we encode them in terms of some spatial frame of reference [180, 146]. Experiments conducted by Mou and McNamara [146, 148] strongly suggest that these intrinsic reference systems are derived from properties of the objects and their surroundings. Explicit rectangular boundaries, such as the walls of a room or the edges of a table, can generate a frame of reference, but a grid-based item layout can also generate an implicit axis of reference. Mou and McNamara [146] showed that people can remember item locations more accurately when they view an object layout from an axis-aligned viewpoint, even if locations were originally learnt from a non-aligned viewpoint.

There is clear evidence showing that boundaries can enhance, but also bias, the accuracy of spatial recall. Nelson and Chaiklin [151] asked participants to recall the location of a dot relative to an enclosing frame border. Recall accuracy was higher the closer the dot was to the border, and inaccurate guesses tended to err in the direction of the nearest border, rather than towards the centre. Igel and Harvey [97] found that the presence of a visual frame increased accuracy overall when participants were asked to remember multiple dot locations. These results suggest that spatial recall in command interfaces may be enhanced by the presence of visual boundaries.

Huttenlocher et al. [95] proposed that since spatial memory is inexact, people increase their accuracy using a two-level encoding which contains both a fine-grained location and a higher-level spatial category (which is determined by the structure of the stimulus). When asked to reproduce exact locations, they will combine the two values. Huttenlocher et al. produced evidence of this in a task which required users to reproduce the location of a dot within a circle: participants spontaneously divided the circle into four quadrants, and used the implicit borders to improve their recall accuracy [95]. The idea of a two-level spatial encoding was further developed into a predictive model of location memory by Lansdale [122].

Egocentric frames of reference, where object locations are encoded in relation to the position and orientation of the body, are also possible. These are typically used to remember locations which are invariant to body position and orientation – one example might be a person who keeps their house keys in their left-side trouser pocket. Conscious manipulation can alter what kind of reference frame is used: Mou et al. [147] demonstrated that in an augmented reality environment, users were able to change from an environmental to an egocentric frame of reference after experience and instruction.

2.4.2 Landmarks

Like frames of reference, landmarks provide a kind of spatial anchor in relation to which other locations may be defined. However, the information provided by landmarks is usually not as rich – a landmark normally serves to identify a single point in space, rather than imply a set of axes through its structure. In the traditional sense, the word “landmarks” normally refers to unique human-made or

natural structures; more generally, however, landmarks can be generated by any visually salient or memorable element of a scene [187].

In many situations, landmarks are extremely important to the development of spatial memory. Allen [4] states that "individuals use landmarks as organizing features within the context of spatial events" (p. 629), and showed that subjects performed much better at a distance-estimation task after being shown images of a landmark-heavy journey. Siegel and White's [182] three-tiered model of landmark, route, and survey knowledge suggests that landmark knowledge is the first type of spatial knowledge to develop when exploring a new environment. Indeed, there is substantial literature highlighting the importance of landmarks in the navigation domain, which is not reviewed here – instead, the interested reader should refer to Freksa and Mark [187].

For table-top object layouts, Mou et al. [148] showed that locations are encoded in terms of inter-object relationships. This suggests that landmarks may be useful in HCI as a way of enhancing spatial knowledge of computer interfaces. One example where landmarks play an important role is in pan-and-zoom interfaces, where they serve as a way to orient oneself (see Section 4.2.1). This effect is most noticeable in situations where landmarks are absent (such as when a map interface is zoomed in too far), resulting in "desert fog" [105]. Further applications of landmarks in HCI are discussed in Section 4.2.3.

2.5 Summary

This chapter provided a high-level summary of some relevant theories of spatial memory, including working memory, long-term memory, and systems of spatial reference. These theories are intended to provide the groundwork for the following two chapters, which look at the empirically observable properties of spatial memory, and interfaces that affect spatial memory, respectively.

Chapter 3

Observable Properties of Spatial Memory

While Chapter 2 focused on psychological theory, the aim of this chapter is to summarise the empirically observable properties of human spatial memory, with an emphasis on those aspects most relevant to human-computer interaction. Extensive prior research, both in psychology and in HCI, has examined and identified human capabilities for spatial memory; this chapter summarises and distills key findings from that research.

Specifically, six properties of spatial memory are identified and examined. First, literature on the *time* to retrieve item locations from memory is discussed. Since re-finding previously seen objects often involves elements of both memory and visual search, this section also reviews literature on visual search.

The second section examines the *accuracy* with which people can remember spatial locations in the absence of feedback, as well as the effects of frames of reference on a person's ability to recall locations. Note that there is a blurred distinction between the literature on retrieval time and accuracy, as there are several ways in which the two attributes interact. For example, as a user improves their recall accuracy for a given item location, the time taken to find that item decreases (since less visual search is required); conversely, if a user attempts to guess target locations as quickly as possible (thus reducing the time available to query their spatial memory), their overall accuracy will decrease.

The third section discusses *effort*. This includes both the effort of memorisation and the effort of recall, though the focus is on the differences between automatic and effortful spatial learning, linking back to the theories presented in Chapter 2.

The fourth section discusses *retention* – in other words, the capacity of the human brain to remember spatial locations, and the length of time these memories persist. As in other sections, empirical results are framed in terms of the theories

presented in Chapter 2 on working memory, long-term memory, and forgetting.

The final two sections look at *variation* in human ability, as well as people's *perceptions* of their own ability and how they can differ from reality.

3.1 Retrieval time

As spatial memory develops, users become more efficient at locating controls. A novice with poor spatial knowledge will rely on slow visual search to find items, while a practiced user can quickly retrieve item locations from memory [33]. The mechanisms by which these skills develop were discussed in Section 2.3; this section discusses user performance in visual search and spatial retrieval. Note that although the process of visual search is distinct from the processes involved in spatial memory, it is important to review elements of visual search because novices must use visual search as the first step in learning item locations. Once stable spatial locations have been learned, visual search becomes unnecessary.

3.1.1 Visual search

Visual search has proven to be a difficult process to model, with empirical studies often producing conflicting results. Specifically, there has been little consensus on whether humans attend to items sequentially, randomly, or in parallel; however, regardless of the underlying process, the average time required for visual search tends to be a linear function of the number of candidate items [93, 155, 176].

This section discusses visual search processes, as well as other factors (such as orderings, groupings, and saliency effects) that affect people's ability to find items.

Is visual search random, sequential, or parallel?

Early work by Krendel and Wodinsky [115] and Engel [51] reported experimental results suggesting that visual search is completely random (i.e., people examine locations at random until the target is found, possibly revisiting already-visited items in the process). Both experiments were performed using random two-dimensional arrangements of stimulus items, unlike the linear command lists used in modern-day menus or grid-based arrangements of toolbar icons. Card

[20] was the first to study visual search in mouse-based linear menus. In a menu of randomly-arranged items, his results (for a small three-user study) showed no difference in search time for different positions in the menu, providing further evidence for randomly-ordered search.

In contrast, Lee and McGregor [124], in their examination of optimal menu structures, proposed a sequential model of visual search where items are examined in top-to-bottom order. In a follow-up paper [132], they re-examined Card's assumptions, and determined that his empirical data could be equally well described by a sequential model. They concluded that either model may be possible, and that users may employ different visual search strategies depending on the application.

Furthermore, Nilsen [155] presented an experiment showing that visual search for individual items was a linear function of that item's position in the menu, strongly suggesting that participants were performing a sequential, top-to-bottom search. Hornof and Kieras [93] and Byrne et al. [19] further investigated this data and derived more complex models, each involving elements of both sequential *and* random search; Hornof and Kieras also provided a key result showing that users do not necessarily consider each menu item individually, and in fact can evaluate several items in parallel if the items can simultaneously fit into the fovea [93].

Importantly, however, all three models of search (sequential, random, and partially parallel) provide the same observable relationship: average search time in a menu is consistently a linear function of the number of items [93, 155, 176].

Note that while some researchers have suggested applying the information-theoretic Hick-Hyman law [86, 96] to visual search (e.g., [188, 210]), others have argued that this is erroneous [177, 29] because the Hick-Hyman law (further discussed in Section 3.1.2) models decision time, not search.

Orderings and groupings

Card [20] followed up his initial menu search work with a second experiment investigating the difference between menu arrangement schemes. His results showed that an alphabetical layout enabled faster visual search than random or categorical layouts, and that a categorical layout was faster than random. How-

ever, the advantage of an alphabetical layout only generalises to situations where users know the command name in advance, and can therefore anticipate the position of the command in the list. Furthermore, the categorical layout may have been unfairly disadvantaged by the fact that Card's implementation did not display labels for each category, forcing users to attend to the items within each one in order to infer what the category was. In any case, differences between the three conditions disappeared on the final block, once users were sufficiently familiar with item locations. Somberg [186] reports very similar results, with alphabetic and frequency-based arrangements enabling faster search initially, but with positionally-constant menus proving significantly faster after several blocks of trials.

In a further study, Hornof [92] compared visual search strategies in grouped menu layouts, investigating both labelled and unlabelled groups. When groups were unlabelled, the partially-random search strategy proposed by Byrne et al. [19] provided the best fit to the data. With labelled groups, users were able to perform a faster hierarchical search: first searching for the target group name, and then searching within that group for the target item. However, such efficient hierarchical searches can only be conducted when the user can predict the name of the relevant group in advance. To compound the issue, generating meaningful category names for commands in real-world applications can often be a difficult task. Nevertheless, in situations where category labels are present, Hornof's results suggest that they should be visually emphasised.

Visual saliency and pre-attentive effects

For visual search in images or structures more complex than linear menus, no models have been developed that accurately characterise search processes. While some researchers (e.g., [100, 161]) have proposed visual saliency-based models of search, in which the eye fixates on visible features in decreasing order of saliency, empirical tests of these models have given mixed results [100, 84]. The order in which features are attended to during visual search appears to depend heavily on the nature of the task [168], as well as complex cognitive factors such as the subject's semantic knowledge of the information being viewed [84].

However, one aspect of visual search is somewhat better understood: the con-

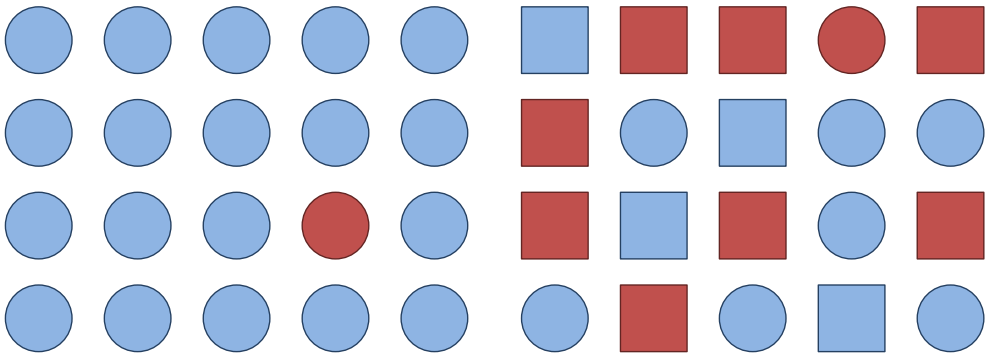


Figure 3.1: In the left image, the single differentiable feature (colour) allows the red circle to be found pre-attentively. In the right image, the conjunction of features (both colour and shape) requires a slower, more sequential search.

cept of *pre-attentive* or parallel search, which can be performed without the need to sequentially scan through individual items [200, 199]. Pre-attentive search occurs when a subject is instructed to find an item based on one differentiating *separable feature* of that item, where the separable feature may be a quality such as colour, brightness, or shape; but does not occur when items are differentiated by a conjunction of features. For example, finding a red circle in a field of blue circles can be done pre-attentively, but finding a red circle in a field of red and blue squares and circles can not (Figure 3.1).

The ability to exploit pre-attentive search to decrease retrieval time, however, relies on the subject knowing in advance that their intended target is going to have a particular differentiating feature [69]. In HCI, this means that visually distinctive icons can exhibit 'pop-out' effects if the user knows what they are looking for. However, artificially encouraging pre-attentive search can be difficult – for example, highlighting likely items with a distinctive colour will only create a pre-attentive effect if the user expects their target to be in that colour (see Section 4.3.3 for more on adaptive highlighting techniques).

One technique that has been shown to cause an *involuntary* attentional shift is *abrupt onset*, where one item appears instantly and others appear after a delay [211]. Yantis and Jonides showed that the involuntary capture of attention caused by abrupt onset was not replicable using salient features (such as colour) [104]. In HCI, Findlater et al. used this idea to speed up visual search with their *ephemeral*

adaptation system (see Section 4.3.3).

3.1.2 Spatial retrieval

While visually searching for items is usually a linear process, repeated searches for the same items allow users to naturally develop a memory for the locations of these items (assuming they remain spatially stable). Measuring the speed of item retrieval from spatial memory during in-the-field computer use is a difficult task – for example, it is unclear when spatial retrieval ends and when mechanical target acquisition begins. For this reason, retrieval time is usually examined indirectly.

Generally, it is accepted that the time to retrieve a spatial location from memory is a logarithmic function of the number of items. Sears and Shneiderman [176] observed that menu selection times increased linearly with menu length for unfamiliar items, but only logarithmically for items that were frequently selected. Cockburn et al. [33] suggested that these expert selections could be accurately described by the *Hick-Hyman Law* of choice reaction time [86, 96], which states that the time taken for a human to make a decision is proportional to the information content of that decision (which in turn is a logarithmic function of the number of available choices).

Cockburn et al. developed and empirically validated a model of menu selections that incorporates linear visual search, Hick-Hyman logarithmic decision making, and the transition from novice to expert behaviour. The model incorporates Newell and Rosenbloom's assertion (see Section 2.3.2) that as users become familiar with selecting commands, their task times improve following a power law of practice [152].

Importantly, if users have perfect spatial knowledge of items, item arrangements designed to shorten visual search times will no longer provide any benefit. Card's 1984 study [20] demonstrates this effect, with differences in search time between alphabetical, semantic and random layouts disappearing once users had learned item locations. In practice, however, location knowledge is rarely perfect (see Section 3.2), and expert users are unlikely to know the locations of *every* command in an application, so techniques designed to reduce visual search may still be useful.

3.2 Accuracy

As discussed earlier, retrieval time is dependent on the user's underlying knowledge of target locations, and better knowledge allows users to avoid visual search. It therefore follows that the reduction in task time with practice, which follows a power law, is simply a manifestation of an underlying improvement in the user's spatial knowledge.

In HCI experiments, the accuracy of spatial knowledge is usually measured indirectly through task time, which is arguably the more important dependent variable. When experimenting on a standard visual interface, it can be difficult to differentiate between visual search and spatial recall behaviours in users. However, a few studies have investigated spatial accuracy directly, using 'invisible' interfaces that lack visual feedback, and measuring accuracy as the dependent variable rather than selection time. These studies are reviewed here.

When visual (or other) feedback *is* present, item locations can be encoded in terms of landmarks and frames of reference (as discussed in Section 2.4.1). The presence of a frame of reference can contribute substantially to a user's ability to recall spatial locations, allowing them to find familiar objects from different viewpoints. Empirical results relating to frames of reference are therefore also discussed below.

3.2.1 Recalling the locations of invisible items

One way to measure the accuracy of spatial recall, without using selection time as a proxy, is to remove all visual feedback and ask subjects to point at the areas where they believe items to be located. While this kind of experiment has been performed many times in the psychology domain, the generalisability of such experiments is heavily dependent on the circumstances of the experiment, the amount of learning taking place before the recall tests, and other factors that are difficult to control. To narrow down the range of results, only invisible-object studies in HCI are considered here, and care is taken to identify the amount of prior learning before recall in each case.

One way to measure typical levels of spatial knowledge is to evaluate participants' existing expertise with applications they use frequently. Gustafson et al. [73] performed such a study on mobile phones, showing that when regular iPhone

users were asked to recall the locations of applications on their home screen, 68% of locations were recalled correctly.

For 3D interaction, Li et al. [126] investigated users' spatial and proprioceptive ability to replicate angular directions, mapping application commands to a spherical co-ordinate space in a 180 degree hemisphere in front of the user's body. To minimise the effect of errors, they recommended dividing the space into 7 horizontal and 4 vertical partitions. In this case, participants were not trained – they were simply given angular directions and had to point in those directions while blindfolded.

Cockburn et al. [28] studied three similar 'air pointing' systems: aiming the pointer at a virtual screen, positioning the pointer within a virtual 2D plane, and positioning the pointer within a virtual 3D volume. In their experiment, participants first completed 48 training trials (12 blocks of 4 targets) with full visual feedback, before interacting with conditions providing progressively reduced feedback. Pointing within a 2D plane enabled the highest accuracy, with ray-casting quick but inaccurate and 3D volume pointing slow, inaccurate, and requiring the highest cognitive and physical effort.

3.2.2 Frames of reference

Section 2.4.1 reviewed literature showing that spatial information is often encoded in terms of a frame of reference. The air-pointing studies in the previous section examined recall accuracy in egocentric frames of reference without feedback – this section discusses the manipulation and creation of visual reference frames in order to enhance spatial memory.

Hinckley et al. [88], studying the manipulation of virtual invisible objects, found that using the non-dominant hand to create a proprioceptive frame of reference increased users' accuracy in a memory task. Based on this idea, Gustafson et al.'s *imaginary interface* system [72] required users to form an L shape with their non-dominant hand, generating an explicit reference axis relative to which the other hand could interact. They found that use of the non-dominant hand in this way partially alleviated effects caused by disorientation (such as physically turning around between tasks), and that interaction accuracy in a co-ordinate pointing task was relative to the Manhattan distance from the user's fingertips.

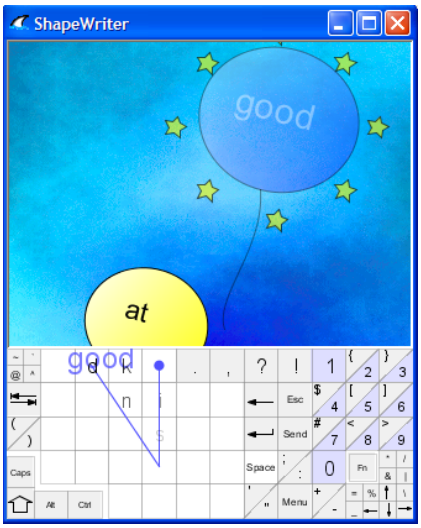


Figure 3.2: Cockburn et al.'s gesture keyboard training game (from [34]). Users must temporarily brush virtual 'frost' off the keyboard with a stylus in order to see underlying key labels.

3.3 The role of effort in forming spatial memories

Section 2.3.3 discussed psychological theory implying that certain types of spatial information are learnt automatically, but that effortful learning can provide benefits in terms of long-term retention.

In HCI, benefits of effortful spatial learning have been observed in training interfaces. Cockburn et al. [34] developed a training interface for a virtual gesture keyboard that required 'frost-brushing': that is, in order to view the labels on the keys, users had to brush away virtual 'frost' with a stylus, which re-formed soon afterward (Figure 3.2). In this way, users were forced to attend to and memorise key locations, in order to avoid having to brush away the frost every time a word was typed. The effortful training was shown to be a more effective use of training time than simple repetition, although the authors cautioned that an excessive level of difficulty may cause users to become frustrated and give up.

Ehret [48] showed a similar effect in a simple item-selection experiment. He examined the repeated selection of buttons representing colours, and manipulated the meaningfulness of the button labels: the most meaningful was a display of the colour itself, followed by the name of the colour, an arbitrary icon, and no feedback at all. Ehret found that the colour-display condition enabled the fastest

interaction, but performed significantly worse than the other three in a spatial recall test. Ehret concludes that "locations are learned more quickly when the least-effortful strategy available in the interface explicitly requires retrieval of location knowledge" (p. 1).

Experiments like Ehret's imply an interesting trade-off between enabling low task times and encouraging long-term learning. For example, interfaces can speed up visual search for items by highlighting a subset of the most likely selections [57, 59], or through the use of visit wear [183], yet there is evidence to suggest that doing so decreases spatial learning, since users become less reliant on their spatial memory [183]. Similar results have been shown in the wayfinding literature, with the presence of GPS devices decreasing users' navigation ability [99]. The best approach for designers therefore depends on the goal of the system: if the explicit aim is training, then requiring more effort can be beneficial; if the aim is simply to allow faster interaction, then making the interface *less* effortful (e.g., by adding highlighting) is probably the better approach (although problems can arise when the highlighting is removed).

3.4 Retention

As discussed in Chapter 2, cognitive psychologists generally divide the concept of memory into two separate systems: short-term (or working) memory, and long-term memory, each with substantially different properties. Measurements of retention, such as how much spatial information can be remembered, and how long it can be remembered for, are therefore highly dependent on which aspect of memory is being utilised.

To re-iterate from Chapter 2, working memory is known to have a limited capacity, with information fading over short periods of time, while the capacity and duration of long-term memory is significantly higher, but more difficult to study. Here, empirical literature from psychology and HCI on the capacity and duration of both working and long-term spatial memory is examined.

3.4.1 Short-term retention

In general, it is known that short-term visual and spatial memory is limited [163, 8, 72]. In Postma and De Haan's study of short-term object location memory, the ac-

curacy with which subjects could remember object locations diminished as more locations had to be simultaneously remembered [163]. Participants in Gustafson et al.'s study [72] demonstrated reduced accuracy in drawing long, complex gestures, suggesting that participants' visuospatial memory for the locations of earlier strokes was fading over time. Baddeley states that the capacity of visual working memory is typically limited to three or four items [8].

Attempts to measure the duration of short-term spatial memory have had varied results. Thomson [197] and Steenhuis and Goodale [189] asked participants to walk to nearby target locations with their eyes closed; the two studies found that target accuracy appeared to decrease after 8 seconds and 30 seconds, respectively, of walking with no visual feedback. Similar experiments in manual aiming have shown that accuracy begins to deteriorate after only 2 seconds without visual feedback [49]. While the duration appears to be highly variable, it is clear that people are able to hold recently-seen visual imagery in their heads at least for a few seconds.

Hole [90] investigated decay effects in visuospatial memory, asking subjects to determine which of two sequentially presented dot pairs had the greater degree of separation. Accuracy decreased linearly as the delay between presentations was increased from 0 to 30 seconds (i.e., a longer delay required a larger difference between the two dot pairs in order for subjects to choose the correct one). They also demonstrated interference caused by extraneous visual information: when a distractor dot pair was presented between the two stimulus pairs, subjects' spatial accuracy was halved.

3.4.2 Long-term retention

While little research has been done to specifically determine the capacity of long-term spatial memory, anecdotal evidence from daily life (e.g., the sheer number of routes and object locations that we accumulate over the course of our lives) suggests that it is very high. Some studies provide hints to the scale of our spatial memory, although they do not identify upper bounds: Jiang et al. [102] investigated spatial contextual memory, showing that people have a large implicit memory for contextual cues. Across the course of a week, they trained participants to find T-shaped targets amongst 60 repeated displays filled with L-shaped dis-

tractors, intermixed with 1800 homogeneous non-repeated displays. Participants were reliably able to locate targets much faster on the displays that had been seen before, even though all displays were highly similar.

In the HCI domain, Robertson et al. studied long-term memory for document locations in their Data Mountain system [169, 39]. In their first experiment, participants placed and interacted with a set of web-page thumbnails on a virtual inclined plane. Six months later, they were given five minutes to review their old layouts, and were asked to find items again; they displayed a similar level of performance to that of the first experiment. However, it is unclear to what extent participants were simply using the five minutes to re-learn item locations.

3.5 Variation in ability

When it comes to spatial tasks, as with many other human capabilities, abilities can vary widely between individuals. For example, Hegarty et al. [83] had subjects walk a route through a building, then asked them to point in the direction of landmarks that were encountered during the route and estimate distances between them. The highest-performing participants "could point to unseen landmarks in the environment with less than 10° of absolute error" and "showed an almost perfect correlation between their estimates of distances among landmarks and the true distances", while the lowest-performing participants had scores "not significantly different from chance" (p. 173).

Certain demographic factors have been found to have effects on spatial ability. For example, there are well-documented differences between the sexes in certain types of spatial tasks. In general, women tend to have better memory for object locations than men [46], while men tend to be better at aiming and throwing projectiles [77, 207], as well as mental rotation tasks, spatial visualisation (e.g., paper folding tests), disembedding (finding a simple figure hidden in a more complex one), and field independence (e.g., perceiving and drawing absolute horizontal and vertical slopes). The interested reader should refer to [78] and [113] for complete reviews of sex differences in spatial tasks.

Light and Zelinski [127] reported an experiment in which adults aged 18-35 were better able to reconstruct a map of town landmarks than adults aged 51-80, suggesting a deterioration of spatial memory with age. However, other studies

investigating the effects of age have had mixed results (e.g., [140, 160]), with differences possibly explained by the visual distinctiveness of stimuli used in the experiment [178].

Cultural differences can also lead to differences in spatial ability. Kearins [110] found significantly stronger visual spatial memory in Australian Aboriginal children compared to Australian children of European descent, though it was unclear whether the difference was due to natural endowment, or simply due to differing child-rearing practices between the two cultures. Similar cultural differences have been found between Scottish and Zairian children [15], and Japanese and Caucasian adults [64], although the latter study takes care to point out that experiential factors may have a large effect, and cautions against over-generalisation.

These individual differences in spatial ability imply a risk for interface designers, which is that interfaces designed to exploit spatial memory may be less effective, or even detrimental, for users with poor spatial ability. However, more research is needed in this area.

3.6 Perception of ability

People perceive their spatial abilities to be lower than they actually are. In Cockburn and McKenzie's comparison of 2D and 3D interfaces [30], participants predicted low performance in a spatial retrieval task, but rated their performance significantly higher after the task was complete, implying an initial lack of trust in their spatial memory. Andrade and Meudell [6] reported a similar phenomenon: many of their participants reported that they guessed their way through a spatial memory test, even though they achieved recall scores that were significantly above chance. This may stem from the automatic, rather than conscious, nature of spatial contextual memory [102].

While in the case of spatial memory, people may be unaware or mistrusting of their own capabilities, it is worth noting that in other situations people will often rely on their (potentially erroneous) memory rather than consulting documentation. Gray and Fu [70] performed an experiment in which participants were told to program a VCR, with reference materials to verify that their actions were correct. When there was even a slight perceived cost to accessing the reference information (such as a single click being required to open it), participants tended to

rely on the imperfect knowledge in their head, rather than verifying their actions against the correct information given to them.

3.7 Summary

This chapter examined six observable properties of spatial memory: retrieval time, accuracy, effort, retention, variation, and perception. Overall, the literature suggests that human spatial memory is highly capable, with quick retrieval times, high levels of accuracy for interfaces that are used frequently, and long memory duration. However, individuals can vary significantly in ability, and people tend to underestimate their own capabilities. Furthermore, the amount of effort involved in learning spatial locations affects the durability of the memories formed, with higher levels of effort resulting in better retention.

Chapter 4

Interfaces and Spatial Memory

The preceding chapters were concerned with the psychological theories and empirically observable properties of spatial memory, and drew information mainly from cognitive psychology literature. This chapter focuses specifically on HCI, reviewing systems and interfaces that support or disrupt spatial memory in some way, and distilling lessons learnt from the evaluations of these systems.

When considering spatial properties of user interfaces, it is possible to divide UIs into two classes: those that display an entire information space at once (for example, a file browser window with only a small number of files), and those that show only small portions at a time (required when the window contains too many files to be displayed together). This chapter therefore first considers situations in which it is possible to simultaneously display all of the items in an information space, the benefits of such an approach, and the applications of such spatial displays to desktop interfaces. When this approach is not feasible due to space restrictions, designers must either provide a user-controllable *viewport* to the space, or somehow *distort* the space such that the information can fit on the screen, with each approach affecting spatial memory differently. This chapter looks at the different ways this can be achieved, and examines how these techniques support or hamper the user's spatial abilities. Additionally, the effects on spatial memory of various types of non-visuospatial cues and feedback are examined.

4.1 Single-view spatial displays

In 2D interfaces, such as those displayed on standard computer monitors, any data to be displayed is necessarily mapped to 2D space. When this mapping is constant, users naturally learn locations as a side effect of use, increasing their performance to a high level over time [107, 129, 48, 33]. However, techniques that dynamically alter the mapping of data to screen co-ordinates, such as scrolling, can re-

duce users' ability to remember item locations; and interfaces that are designed to manage complexity, such as hierarchical menus, can limit the usefulness of spatial memory by requiring the user to perform comparatively slow mechanical actions to select items.

These two problems of spatial unpredictability and suboptimal mechanical selection can be solved by displaying all of the items in an information space at once, in a non-overlapping spatially consistent layout (see Figure 4.1 for an example). Displays such as these reduce mechanical navigation time compared to scrolling lists [74], and allow users to utilise their spatial memory to quickly access items they have encountered before [74], as well as providing an instant overview of the information space [157, 32], among other benefits. This section investigates some of the research interfaces in HCI that follow this principle of simultaneous presentation, and show how they affect task times and spatial memory in comparison with traditional methods of interaction.

4.1.1 Advantages of spatial layouts over scrolling

Several research projects have shown the value of spatial layouts for working with documents compared to more traditional techniques such as scrolling. In 1997, O'Hara and Sellen [157] studied differences in reading behaviour between paper and on-line documents. When interacting with paper, all participants unclipped the documents they were given and arranged them spatially on the desk, citing advantages such as quick cross-referencing and "gaining a sense of overall structure" (p. 339). The authors noted that digital techniques for managing documents, such as scrolling and paging, were "irritatingly slow and distracting" (p. 338), and that they significantly reduced the user's ability to use spatial memory for navigation. In a follow-up paper [158], they developed a focus+context reading interface which showed an overview of the whole page, and displayed one full-size sentence at a time. The interface with the overview proved to better support incidental learning of spatial locations than a standard scrolling interface.

Similar results in favour of spatial overviews in document navigation were demonstrated by Cockburn et al. [32]. Their *Space-Filling Thumbnails* system allowed users to view a spatially consistent overview of a document, in which thumbnails of every page were arranged in a 2D layout and scaled such that all of

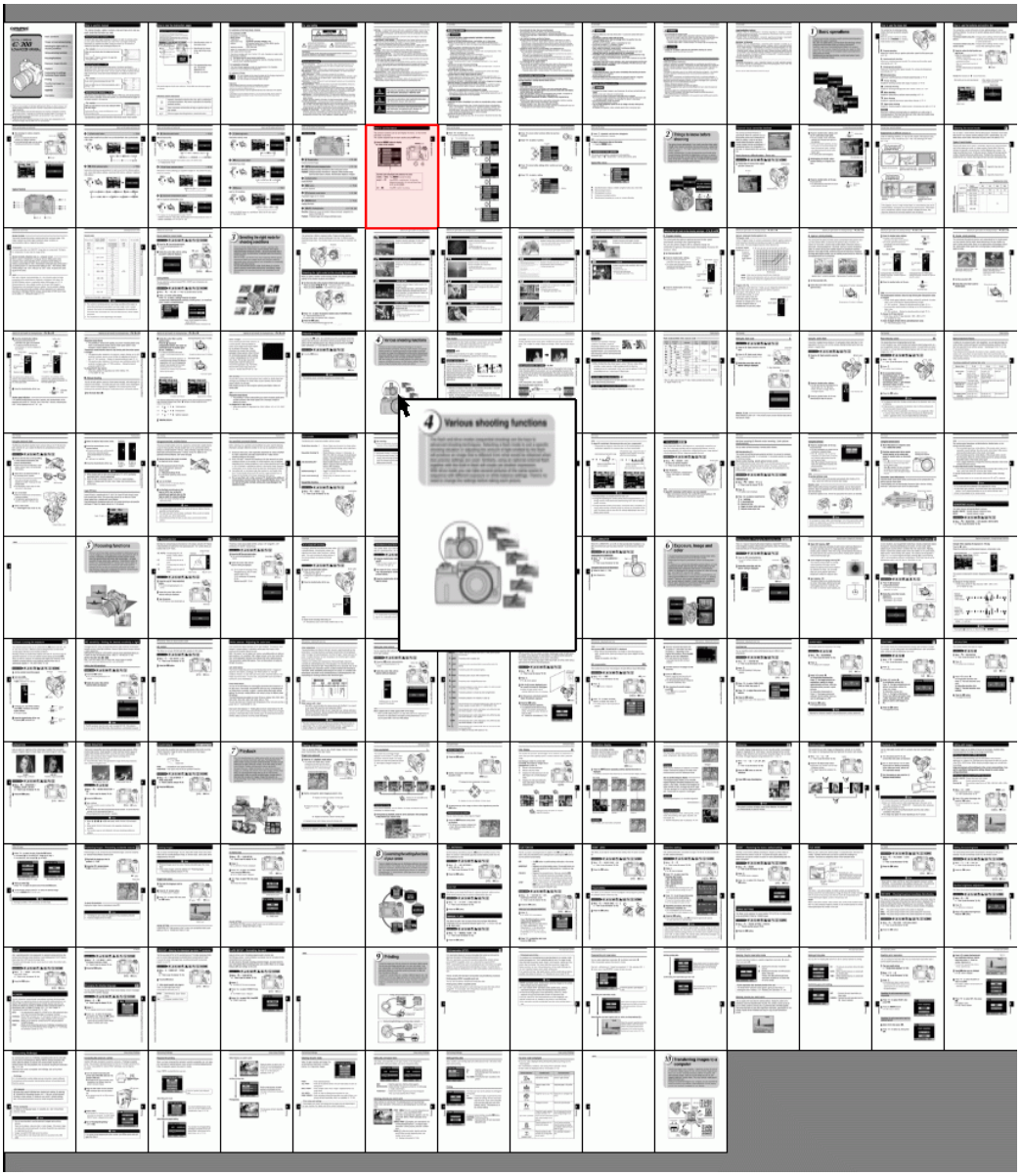


Figure 4.1: Space-Filling Thumbnails allow an entire document to be displayed on a single screen, using a spatially stable arrangement. Image from [32].

Figure 4.2: A ListMap displaying a list of system fonts. Image from [74].

the pages could fit in one window (Figure 4.1). Space-Filling Thumbnails were shown to be better than six other document navigation methods in terms of both visual search and retrieval of locations from spatial memory, with spatial retrieval being particularly quick [32].

Spatial arrangements have been shown to be beneficial in many other application domains. Tak et al.'s Spatially Consistent Overview Thumbnail Zones (SCOTZ) [191, 192] provides a full-screen, spatially stable interface for window switching, allowing quick revisitation. SCOTZ divides the screen into application zones that grow based on the frequency of use. The problem of dynamic data sets, as well as changing zone sizes, is alleviated by the use of spatially stable treemap algorithms [202, 190], which keep items as close as possible to their original locations when the data set changes. SCOTZ showed significant advantages over existing window switching tools [192].

Gutwin and Cockburn's ListMap design [74] laid out a list of items in a 2D grid that required no scrolling (Figure 4.2). Task times with ListMaps were significantly lower than with the scrolling list for revisitation, but significantly higher for users who had to find items for the first time. However, their interface constrained item sizes to 23×14 pixels, and used only textual item names (which required a mouse-over to view in full), making visual search costly and time-consuming.

Robertson et al.'s Data Mountain [169, 39] was a document management system that allowed users to arrange their documents on a 3D inclined plane. In a

study using web-page thumbnails, the spatial arrangement of items proved to be more efficient than Internet Explorer's Favorites system (a scrolling list), although the authors did not separate the relative contribution of the third dimension, the thumbnail images, the ability of users to organise and cluster items according to their preference, and other factors.

Later, Cockburn and McKenzie [31] isolated and studied the 3D aspect of Data Mountain, comparing it to a 2D equivalent as well as a real-world 3D interface (in order to avoid the technological limitations of virtual 3D). Their study showed no spatial memory benefits for the 3D interfaces (failing to replicate an earlier study by Tavanti and Lind [196]), indicating that the main strength of Data Mountain was its 2D spatial layout, which enabled users to remember items by location. This idea is supported by results in psychology – Shelton and McNamara [179] found that memory for 3D scenes was strongly viewpoint-dependent, suggesting that people may not automatically encode three-dimensional information.

4.1.2 Spatial layouts and menu hierarchies

As shown in the previous section, spatial arrangements support quick revisitation; and when spatial knowledge of an interface is strong, reduction of selection times tends to be bottlenecked by navigation costs such as scrolling or navigating hierarchical menus. However, hierarchies serve specific benefits, particularly for novice users [89], in that they reduce information load and provide a categorical structure to guide the user's search for commands [172]. This naturally leads to the question: how deep should hierarchies be?

There has been considerable investigation into how best to design menu hierarchies, with literature on the appropriate breadth and depth of menu systems dating back to Miller [143]. Miller's initial experiment evaluated different hierarchical arrangements of 64 menu items, ranging from one broad menu containing all 64 items to a tree of depth 6 with two choices at each level. His results showed a U-shaped function for selection time, with the two-level, eight-choice-per-level layout the most efficient; these findings were replicated in similar studies by Snowberry et al. [185] and Kiger [112]. However, other studies, such as that of Landauer and Nachbar [120], showed benefits for flatter hierarchies, contradicting Miller's results.

The difference appears to lie in the predictability of item locations, either by inference based on an understanding of the layout (such as an alphabetical order) or by pre-existing spatial knowledge. Miller's menu items were randomly ordered, while Landauer and Nachbar's were ordered alphanumerically, enabling the user to anticipate the locations of the items. Based on this, Cockburn and Gutwin [29] developed a predictive model that explained and integrated the results of previous studies. They demonstrated that certain types of data organisation (such as alphabetic or categorical ordering), as well as expertise with spatial locations, allow for logarithmic search time; and that this kind of search favours broad, shallow hierarchies. Conversely, visually searching data that is structured unpredictably is aided by a small amount of hierarchy, as per Miller's original results [143]. However, interfaces are rarely arranged randomly. Furthermore, it takes only a few repeated selections for users to develop strong knowledge of an item's location (e.g., [33]), suggesting that the "broad and shallow" arrangement is more generally useful.

Recently, there have been investigations in HCI into interfaces that completely flatten hierarchical structures. Parallel dropdown menus, which display all menus on the system simultaneously, are a simple example. Quinn and Cockburn [165] compared fully parallel menus, partially parallel menus (i.e., standard dropdown menus) and serial menus (where only the item under the user's cursor is visible). They found no significant difference between partial and parallel menu types in terms of selection time, although the experiment was of insufficient length to properly examine learning effects or revisitation times. Experiments in web page menus showed favourable results for parallelisation, particularly for experienced users [89]. The design of ExposeHK [136], a system for overlaying hotkey bindings on top of corresponding commands, also suggested the use of parallel menus to display all application commands at once, with a brief analysis showing that modern screens are large enough to display the entire set of top-level menus in most applications.

4.2 Viewports and spatial memory

The previous section discussed the advantages of simultaneous spatial interfaces, which show an entire information space at once. However, this kind of interface is not always feasible, particularly when the size of the dataset is larger than the

display can handle.

One way to solve this issue, while maintaining the internal spatial relationships of the dataset, is to provide a *viewport* to the information space that allows the user to view a manageable subset of the space. This subset is normally controlled through the use of pan and zoom tools (though see Smith and Taivalsaari's "stationary scrolling" [184] for an alternative, spatially constant way to view subsets of data). In panning and zooming interfaces, the user's ability to build an accurate mental model of the space and navigate around it depends on whether or not an overview is visible [18, 158], whether recognisable landmarks and spatial indicators are present [105], and whether constant reference points are available to anchor the position of the viewport [109]. The challenges of pan-and-zoom interfaces are summarised below, and the importance of each of these factors are discussed.

4.2.1 Panning and zooming

Interfaces for viewing large information spaces, such as maps, typically offer two ways to change the view of the data: panning (i.e., translating the viewport), and zooming (i.e., scaling the data).

The ability to zoom and pan in user interfaces creates two challenges for spatial memory. First, the user can no longer learn item locations relative to the boundary of the viewport, since it changes with every navigation action. Instead, the user must learn locations relative to nearby landmarks, in order to mentally assimilate them into a global picture of the space (this process is suggestive of Siegel and White's progression from landmark to survey knowledge – see Section 2.1). This implies that location learning in a pan-and-zoom interface may be less automatic than in a static interface (see Section 2.3.3). It also creates a risk that no known landmarks will be visible, requiring the user to alter their zoom level to determine where they are (the worst case of this effect is known as "desert fog" [105], where the zoom level is so extreme that no landmarks are visible at all). The second challenge is that when attempting to revisit a previously-seen location, the user must first establish the current position of the viewport (again integrating visible landmarks with a mental global picture of the space), before they can determine how to proceed.

Bederson [14] summarises the advantages and challenges of zoomable user interfaces (ZUIs). One of the things he notes is that ZUIs have the potential to "tax human short-term memory [...] because users must integrate in their heads the spatial layout of the information" (p. 4). However, this effect can be alleviated somewhat through animation: Bederson and Boltman [13] showed that the use of animation when zooming enhanced spatial comprehension of an information space. It is worth noting, however, that studies of animation in other areas have provided inconclusive results: see Tversky et al. [203].

4.2.2 Overviews

As noted in Section 4.1.1, spatial layouts of the entire information space make it easy for users to build a spatial model of that space. When using an overview of this kind as the primary view is infeasible, it can sometimes be useful to add a low-detail overview as a secondary window to the data. Overviews of this kind are typically displayed in a separate on-screen location to the primary view, displaying a low-detail version of the entire information space, as well as indicating the location of the primary viewport within that space. Overviews are present in many computer games, such as *Starcraft*, in the form of "mini-maps", and standard scrollbars are essentially simple, one-dimensional overviews.

Intuitively, it seems that an overview should provide benefits for spatial memory and interaction, since they allow the user to anchor the detailed information in their primary viewport to its absolute spatial location. Burigat et al. [18] presented evidence to support this on mobile devices, demonstrating that the presence of an overview in a ZUI increased spatial recall accuracy. Similarly, in mobile reading interfaces, O'Hara et al showed that page overviews support incidental learning of spatial locations in reading interfaces, while pure scrolling interfaces do not [158].

However, some results have shown that overviews are not always useful. Hornbaek et al. [91] found no benefit for overviews in a zoomable map application, with subjects actually performing worse on navigation time and recall accuracy when an overview was present. Nevertheless, a majority of participants preferred the overview interface, making comments like "It is easier to keep track of where I am" (p. 376).

4.2.3 Artificial landmarks and visual cues

Since object locations are generally defined in relation to the positions of other objects (see Section 2.4.2), landmarks are extremely important when it comes to encoding and retrieving location knowledge. In viewport interfaces, landmarks provide another potential method for users to mentally consolidate the data they can see into their overall spatial understanding.

In situations where natural landmarks are not available, dynamically generating artificial landmarks based on user behavior can be useful for supporting revisitation. Alexander et al. [3] applied this to within-document navigation with their Footprints Scrollbar, which places coloured marks in the scrollbar at recently visited document locations. The results indicate that these visual cues can act as landmarks which significantly decrease revisitation time.

Dynamic landmarking has also been applied successfully to two-dimensional interfaces. Skopik and Gutwin [183], in their investigation of fisheye views, noted that revisitation is difficult when distortion causes items to move around, particularly in featureless landscapes. To solve this problem, they added "visit wear" (derived from Hill et al.'s *read wear* [87]) to show previously visited locations, improving revisitation performance. Skopik and Gutwin noted that the risk of this approach is an increase in visual clutter, although this effect can be mitigated by fading out wear marks over time, or allowing the user to customise the visibility of the wear.

Importantly, Quinn et al. [166] found no benefit for dynamic landmarks in a situation where item locations were already easily predictable. In these cases, spatial memory is likely strong enough to find items without the use of visual landmarks. Quinn et al. also caution against the overuse of landmarks, finding that the extraneous visual clutter can distract users and decrease their performance.

In viewports of two-dimensional data spaces, it can sometimes be useful to visualise the locations of objects that are outside the current viewport. City Lights [215], Halo [12], and Wedge [71] are all examples of systems that provide visual indicators of the presence of off-screen objects, with the latter two techniques providing information about both distance and direction.

Another application of such visualisations exists when data is dynamic, or when more detail is desired for several separate locations. In these cases, simple

overview+detail interfaces are insufficient, as it becomes difficult or impossible to simultaneously visualise all of the salient data points. Ion et al. [98] solved this problem for large displays with their Canyon visualisation, which augments an overview+detail interface with a paper-folding metaphor to display the relationships between out-of-view objects and the main focal region.

In summary, artificial landmarks and other visual cues should only be used when existing landmarks are scarce ("desert fog" situations), in order to help a user orient themselves in scrolling, panning and zooming interfaces; or to indicate salient information displayed off-screen. The research indicates that they are less useful in static interfaces, which typically have a frame of reference by which locations can be remembered, and interfaces with many existing landmarks, when adding extra landmarks simply increases visual clutter.

4.2.4 Situated information spaces

In 1993, Fitzmaurice [62] introduced the idea of *situated information spaces*, where a data set is mapped into real-world space, and a spatially aware handheld display can be used as a viewport to visualise the data. Zooming and panning navigation can then be achieved in a natural way, as if looking through a handheld "window" into the virtual world. In terms of spatial memory, the theoretical advantage of this approach is that the user can utilise reference points in the environment to anchor their mental map of the data, reducing the potential for getting lost compared to standard panning interfaces.

However, studies on situated information spaces so far have been limited by the sensing capabilities and response rates of hardware, removing the illusion of the display as a 'window' to the data. Fitzmaurice et al.'s *Chameleon* [63, 61] was the first implementation of such a system. They found that the movable display provided comparable depth perception ability to that achieved with a standard display, but interaction performance was hindered by hardware limitations – however, they did not investigate spatial memory. A later version mounted on a mechanical arm, the *Boom Chameleon* [201], showed benefits for examining 3D objects, although spatial memory was again not measured.

Yee's *peephole displays* [213] applied the Chameleon concept to mobile devices. Yee's prototype showed advantages over a conventional interface for draw-

ing applications, but no difference for map navigation. However, Yee was also affected by hardware limitations, noting that "peephole techniques would work much better on a faster and brighter display" (p. 5).

Pahud et al. [159] studied spatial recall performance using Chameleon-style interaction on a mobile device, and found no significant differences compared to standard 'pinch-flick-drag' interaction.

Recently, peephole displays have also begun to appear on mobile projectors, where the "window" to the data is projected into the environment, rather than being displayed on a handheld device. Kaufmann and Ahlstroem [108, 109] studied map navigation and spatial memory in projected peephole interfaces, and found that they offered better spatial recall accuracy than standard touch-screen panning interfaces for both users and observers of the application.

Rädle et al. [167] studied the use of a tablet computer as a viewport into a wall-sized display. To navigate around the space, they compared standard pan-and-zoom techniques to peephole interaction, which required physical movement of both user and device. Memory for object locations was no different between techniques immediately after the experiment, but a further experiment conducted after 15 minutes of distraction showed increased accuracy in the peephole condition, suggesting that peephole interaction may have benefits for long-term spatial memory.

4.3 Distorting space

Another way to deal with large information spaces is to *distort* the contents of the space such that the most relevant information is more accessible than the rest. This can be done in different ways, depending on the type of data being displayed. In continuous spaces, such as maps, techniques such as pointing lenses can be used to view certain areas in more detail: one well-studied example is the *fisheye view* [65], which allows the user to magnify an area of interest in an information space while maintaining the connectedness of the area to its surroundings (Figure 4.3). With discrete data that does not have an inherent spatial arrangement, such as sets of commands, designers have the flexibility to move and rearrange the discrete parts of the set in any way they choose, in order to make some commands more easily accessible or fit the constraints of the containing interface.

Figure 4.3: Fisheye views seamlessly integrate a high-detail 'focus' region with its low-detail surroundings.

In general, changing the locations of interface objects in this way disrupts a user's ability to use their spatial location knowledge. For this reason, interface design guidelines often promote either spatial stability (e.g., Hansen's 'display inertia' [79]) or the more general concept of 'consistency' [2, 154, 181]. However, there are situations in which system designers may choose to disrupt spatial memory in pursuit of other goals. The rest of this section discusses the situations where this may occur, analyses existing interfaces that distort space and break the principle of spatial consistency, and identifies instances where alternative, spatially-consistent designs can improve performance.

Two categories of spatial distortion are examined. The first category includes focus+context and adaptable interfaces, two situations where the distortion is *user-controlled*. Second, *system-controlled* distortions are examined, such as those generated by adaptive interfaces, and automatic re-arrangement of controls when window geometry changes.

4.3.1 *Focus+context interfaces*

Like an overview+detail interface (discussed in Section 4.2.2), a focus+context interface is designed to provide an overview of the space, while allowing the user to view a specific region (the focus area) in more detail. However, a focus+context display removes the discontinuity between the two regions, "integrating focus and context into a single display where all parts are concurrently visible" ([35], p. 10). The most commonly studied example of a focus+context interface is the *fisheye view*, shown in Figure 4.3.

The fisheye view provides a distortion of visual space while leaving the motor space undistorted (i.e., clicking at the cursor position in the centre of the lens is equivalent to clicking without the lens present). However, despite the fact that the primary goal of a fisheye lens is to enable better understanding of the underlying dataset, these visual distortions can actually increase confusion and disorientation [22] and impair the user's ability to make spatial judgements. This problem is exacerbated when the user has no easy way of mapping magnified items to their positions in the regular, un-distorted space. Zanella et al. [214] reduced these problems in maps by adding visual cues that made the fisheye mapping more explicit, with both a cartographic grid overlay and shading to better indicate the fisheye 'bubble' making it easier for users to comprehend the effects of the distortion.

Skopik and Gutwin [183] examined the use of fisheye views in both maps and discrete graphs, and also found that people had difficulty remembering object locations due to the distortion. Their solution used dynamic landmarks in the form of "visit wear" (see Section 4.2.3) to provide spatial points of reference for users.

4.3.2 *Adaptable interfaces*

In contrast to *adaptive* interfaces, where changes to the interface are determined by the system, *adaptable* interfaces can be directly modified and customised by the user. Findlater and McGrenere [54] compared static, adaptive and adaptable menu systems and found that adaptive menus performed the worst, due to their lack of predictability, while static menus were the most efficient. Participants, however, thought they were most efficient with adaptable menus, even though this was not the case; they also preferred the adaptable menus overall. This suggests

that greater control over an interface can provide users with higher perceived performance and satisfaction.

4.3.3 Adaptive interfaces

Adaptive interfaces are interfaces which "automatically tailor the presentation of functionality to better fit an individual user's tasks, usage patterns, and abilities" [53]. In practice, an adaptive interface typically develops a model of the user's behavior over time, and uses it to alter the interface for the purposes of reducing visual search time or motor movement.

There are several ways in which interfaces can be altered. First, items can be moved: for example, pull-down menus could dynamically re-arrange themselves such that the most frequently used items are near the top (e.g., [145, 176]), enabling lower pointing times for the most commonly accessed targets. Sears and Shneiderman [176] showed efficiency benefits for this type of design in an experiment where item ordering was pre-determined: that is, items were sorted based on *a priori* knowledge of their selection frequencies and did not move during the experiment. However, when menus change dynamically, the cost of unpredictable command locations can outweigh the benefits of the adaptation [145, 54].

There have been several approaches to mitigate this issue. At the algorithmic level, some adaptive algorithms are more stable than others: frequency-based sorting is much more stable than recency-based sorting, for example [33]. Fitchett and Cockburn's AccessRank [58] attempts to create predictive lists that are both accurate *and* stable, although its use in real interfaces has not been explicitly evaluated. Importantly, stability of results is likely to be more important than the understandability of an algorithm, at least in terms of task time: participants in a study by Gajos et al. [68] performed no better with recency-based adaptation than with purely random adaptation (i.e., where controls promoted by the adaptive interface where chosen at random), even though they indicated that the recency-based strategy was much more predictable.

Another alternative is to adapt the interface by *copying* items into an easily-accessible area, rather than rearranging the existing items. Gajos et al.'s Split Interface [67], which copied frequently used items in Microsoft Office into a designated toolbar, was faster than the standard interface and also preferred by users.

The risk of duplicating items is that it may introduce a decision cost – users will not always find the item they need in the predicted subset, so an optimal selection strategy requires them to decide which location to look in first. Howes et al. [94] showed that providing shortcuts to items slows down users for items that have no shortcuts – presumably since participants were checking the shortcut area before looking for items in the standard location.

Some adaptive interfaces preserve spatial layout and make purely visual changes, aiming to accelerate visual search rather than navigation time. Previous work has shown that colour-based highlighting (e.g., [57, 67]) has proven ineffective in aiding visual search, perhaps because of the failure of colour to compete with existing saliency effects in the interface. Solutions that reduce the saliency of non-highlighted items have fared better. Findlater et al.'s *ephemeral adaptation* [57] successfully aids visual search by immediately displaying predicted items, and fading in other items over time; their idea was based on the known pre-attentive effects of abrupt onset (see Section 3.1.1). Fitchett et al. [59] also showed benefits for stencil-based highlighting, which increases the relative visual saliency of highlighted items by adding a dark translucent layer to the rest of the interface.

According to Fitts' Law, pointing times can also be reduced by increasing target sizes. Cockburn et al. [33] investigated *morphing menus*, where frequently used menu items grow in size, but item ordering is maintained. However, they found no benefit over traditional menus. Tak et al.'s SCOTZ window-switching system [191, 192] also increases the size of items over time in a 2D layout, but attempts to reduce the movement of items by using a relatively stable treemap layout. As a result of this, items gradually drift over time, but do not move substantially between invocations of the tool.

Importantly, however, spatial stability is not always the most important concern in an adaptive interface, particularly when screen size is constrained. Findlater et al. [56] evaluated an adaptive split menu with differing levels of accuracy on both small and large screens. Results suggest that the adaptation is of little use on large screens, even at 80% accuracy, but that the high cost of regular navigation on small screens makes the adaptation worthwhile. This result suggests that when navigation cost is low, such as on large screens with shallow hierarchies and rapid input techniques, predictability and consistency (i.e. spatial stability) are the most important factors. When navigation cost is high, e.g. on small-screen

devices using constrained input methods such as rocker switches, consistency is less important and rearranging to improve navigation may be the best strategy to increase efficiency.

4.3.4 Responding to changing display parameters

In general, computer interface layouts are subject to certain parameters, such as the amount of data to be displayed, and the available area in which to display it. If these parameters never change, it is easy to maintain spatial consistency. However, when window geometry changes, or when new items need to be added to the display, interface designers must find a way to adapt the existing spatial layout to accommodate the changes.

The most common approach in these situations is to rearrange existing items – for example, when a file browser window (displaying a tile view) is resized, items are usually reflowed from left to right, top to bottom to fit the new window bounds. Similarly, when new files are added to a folder, they are inserted to maintain the existing order, necessitating a shift in location for all of the items following the insertion point (unless the folder is sorted by creation date, in which case the new file is inserted at the end).

However, in situations where spatial memory is a primary concern, there are alternative techniques available to support these changing parameters. When datasets change, minimizing the change in item locations can be done by adding new items at the end of lists, or using relatively stable visualisations such as treemaps [191, 202, 190]. There are also interesting design opportunities in applying results on frames of reference (Section 2.4.1) to computer interfaces, allowing them to be spatially altered while maintaining the user's memory for item locations. This idea is explored further in Chapter 8.

4.4 Non-visuospatial cues and feedback

The use of spatial memory can often be enhanced by the presence of contextual cues or feedback. For example, Section 2.3.1 discussed how recall of a specific memory can be influenced by cues that were present when that memory was encoded. Similarly, recall may be aided by the presence of feedback that is either non-visual (such as proprioceptive feedback during selection) or non-spatial (such

as text labels on items). This section examines interfaces that provide direct proprioceptive feedback (such as touch screens), identifies the importance of visual and textual cues for confirmation, and discusses the effects of auditory and environmental factors.

4.4.1 Proprioceptive feedback

When a user makes a physical movement to interact with a computer, such as typing on a keyboard or using a touchscreen, they receive proprioceptive feedback. With practice, interactions with the computer can then be encoded in terms of the required motor actions (i.e., using muscle memory), rather than declarative knowledge of item locations.

Tan et al. showed that direct proprioceptive interaction with items (i.e., using a touch screen vs. using a mouse and monitor) can enhance spatial learning, especially for females [194]. The advantages of direct interaction were also shown by Jetter et al. [101], who performed a study showing that touch interaction improved spatial recall compared to the mouse in a panning interface, but not when using a panning+zooming interface. They hypothesised that the changing visual and motor distances induced by zooming negated the proprioceptive benefits of the direct interaction with the surface.

It is also possible that the feedback gained while placing items, such as in file browsers or Robertson's Data Mountain [169], may enhance users' spatial memory for the items they placed. Subjects in the Data Mountain study certainly displayed strong spatial recall, but more research is needed to determine the role of manual object placement in these results.

4.4.2 Visual and textual cues

As discussed in Section 3.2, spatial memory can often be imprecise: users often know the approximate locations of controls they have used, but their knowledge is rarely exact. Combined with the fact that people often mistrust their own spatial abilities (see Section 3.6), this suggests that relying on spatial memory *by itself* may not be an effective method of interaction for most users.

Jones and Dumais [103] performed an experiment where subjects read news articles and virtually filed them into either a spatial arrangement of unnamed cat-

egories, a list of named categories, or a spatial arrangement of named categories. Participants were then given statements about each of the articles, and had to choose the article that each statement belonged to. The named spatial index performed better than the name-only index, suggesting a benefit of spatial arrangement, but the spatial-only index performed the worst by far.

Czerwinski et al.'s follow-up experiment on the Data Mountain system [39] studied the effects of the presence of thumbnail images on spatial retrieval of documents. When the thumbnails were removed, subjects' performance dropped, but returned to normal after a few trials (though mouse-over text was still available for users to verify their selections). Czerwinski et al. observed subjects "homing in on the cluster of web pages they knew to contain the target page, after which they would use the mouse-over text to find the specific target page" (p. 7).

The results of these two studies show that people can use their spatial memory to remember general vicinities in which items are located, but that other feedback (such as thumbnails or text) is required for the user to verify that their intended selection is correct.

4.4.3 Auditory feedback and environmental factors

Some studies have investigated the effects of auditory cues on spatial abilities. Auditory feedback does contain a spatial dimension (when listening to a sound, we can often make a rough estimate of the direction and distance of the source) and this effect is often used in surround-sound systems for movies and games, but results for aiding spatial retrieval in HCI have been mixed. Robertson et al.'s Data Mountain system [169] included auditory cues when document thumbnails were being dragged; controlling volume, panning and reverb levels to indicate spatial location. However, this was consistently ranked as the least helpful cue by participants.

Davis et al. [42] performed another study on the effects of audio in a virtual environment. Participants interacted with virtual rooms, with and without a distinctive auditory cue in each room. They then underwent a spatial recall test where they were asked to remember which objects were in each room, and a recognition test, where objects were shown and participants were asked which room they belonged to. Recognition was more successful when high-fidelity audio was present

(compared to low-fidelity and no audio conditions), and recall was marginally better.

Other environmental factors have been shown to be important to learning. Tan et al. [195] showed that using physically large displays improved users' spatial memory and performance in other spatial tasks. In a separate study, Tan et al. [193] improved the learning of word pairs by creating an immersive environment and placing items in different spatial locations.

4.5 Summary

This chapter presented a review of commercial user interfaces and HCI research systems that affect, or are affected by, spatial memory. This chapter concludes the literature review section of this thesis, which forms the theoretical basis for the systems and experiments presented throughout the remainder of the document.

Part II

Designing Interfaces that Exploit Spatial Memory

Chapter 5

CommandMaps: Exploiting Spatial Memory for Rapid Command Selection

In modern applications, commands are typically arranged in hierarchical structures, such as cascading menus. In 2006, Microsoft released the Fluent UI (or "Ribbon"), which replaced the old menus in Microsoft Office with a tab-based toolbar. The Ribbon allows for a more spatial and graphical command layout (see Figure 5.1), while also permanently displaying the commands from the most recently accessed category. However, both of these systems inhibit performance by forcing the user to navigate a hierarchical structure.

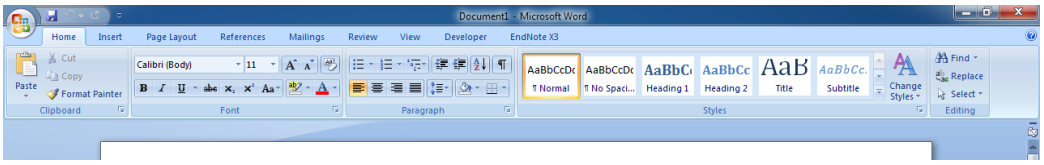


Figure 5.1: Microsoft's Ribbon interface, from Word 2007.

This chapter presents the design, theoretical modelling, and empirical evaluation of the CommandMap, a command selection interface designed to improve performance over traditional interfaces such as menus and Ribbons. The CommandMap design utilises two of the key results presented in Section 4.1, which are (1) that flatter hierarchies allow higher performance for users who know item locations [29, 33, 120], and (2) that single-view, spatially-stable overview displays have been demonstrated to support spatial memory and item revisitation [32, 39, 74, 169, 191, 192].

The CommandMap described in this chapter is a prototype design based on the Ribbon from Microsoft Office 2007. (Real-world implementations of the CommandMap are described in Chapter 6). The CommandMap, instead of displaying

controls in a tab structure, uses the whole screen to display all command categories simultaneously (Figure 5.2). This chapter presents the design rationale behind the CommandMap, mathematical performance models predicting its success, and the results of three laboratory experiments evaluating expert performance with CommandMaps, novice performance with CommandMaps, and different strategies for dealing with variable display space.

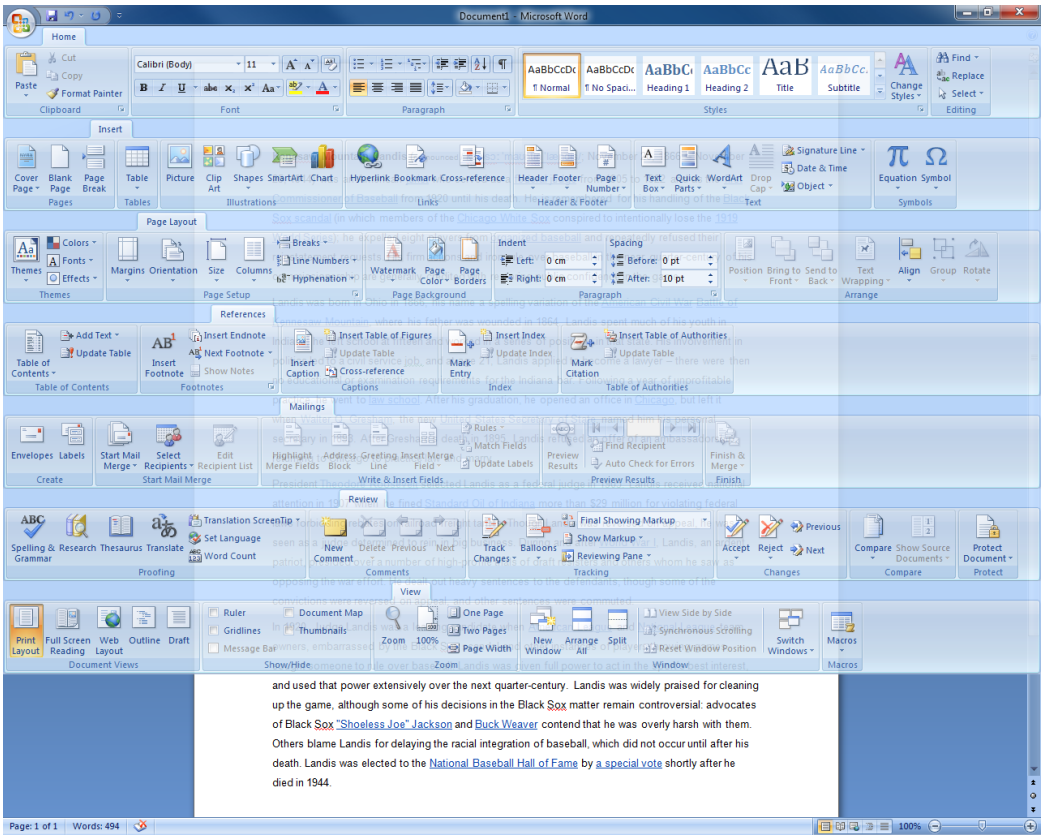


Figure 5.2: A CommandMap prototype based on the Microsoft Office Ribbon, which displays seven Ribbon tabs simultaneously.

5.1 CommandMap design

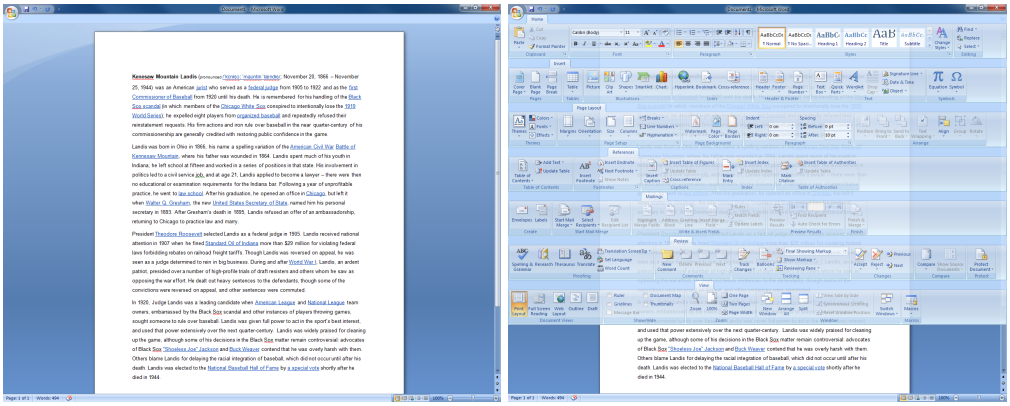
5.1.1 Overview

CommandMap interfaces (e.g., Figure 5.2) are intended to replace traditional command interfaces such as menus, Ribbons and toolbars. Their primary de-

sign feature is that they display all available commands, or as many commands as possible, in a single display. The more the hierarchy is flattened, the quicker it is to access commands; though if an application contains too many commands to display at once, CommandMaps can still contain a small amount of hierarchy (for example, some of the buttons in the Figure 5.2 prototype provide dropdown menus).

Because the interface occupies the entire display space, it is not permanently visible (Figure 5.3). The CommandMap is activated either by an on-screen activation button, or a command key such as `Ctrl`. It then rapidly fades in to a configurable opacity level, allowing the underlying workspace to be viewed. Command selections are then made by clicking on the appropriate icon in the CommandMap. If an activation key was used, the CommandMap remains visible until the key is released, allowing multiple commands to be issued in sequence without reposting.

When the CommandMap is not posted, the entire display area can be dedicated to the workspace (or optionally, a subset of the CommandMap's commands, such as Word's Home tab, can be permanently displayed).



(a) CommandMap hidden (b) CommandMap posted

Figure 5.3: The CommandMap prototype in both hidden and posted states.

5.1.2 Design objectives

The four design objectives of the CommandMap are discussed below.

1. **Compatibility with traditional interaction.** Traditional WIMP interfaces have dominated desktop interaction for thirty years. Although faster command invocation mechanisms (such as shortcuts) are available for experts, it is known that these facilities are lightly used [24, 121] and that most users are content to 'make do' with mouse driven selections. CommandMaps therefore maintain the familiar 'point and click' style of interaction.

2. **Improve performance for knowledgeable users.** The primary objective for CommandMaps is to improve performance for knowledgeable users. Many office workers use the same computing tools for years or decades, and they are therefore likely to be knowledgeable much longer than they are novice. Spatial stability and hierarchy flattening are the two methods by which CommandMaps aim to improve the performance of knowledgeable users.

   Spatial stability. As discussed in Chapter 4, there is extensive empirical evidence showing that consistent spatial placement facilitates location learning and improves selection performance, particularly for controls that are used frequently.

   Hierarchy flattening. Traditional interfaces display only a small subset of commands at a time, so command hierarchies are used to partition command subgroups. The result is that even when users know the ultimate location of their targets, they need to mechanically navigate the command hierarchy to satisfy interface requirements. Furthermore, in tab-based interfaces such as Microsoft's Ribbon, each top-level category constitutes an interaction mode, introducing the risk of mode errors: for example, "Zoom" is not displayed at its known location if the "Home" tab is selected. Previous work discussed in Section 4.1 observes that interface expertise is best supported when interfaces provide a flat command structure. CommandMaps provide a graphical means for hierarchy flattening, maximising the proportion of commands immediately available and reducing the risk of mode errors.

3. **Maintain performance by novice users.** While CommandMaps are primarily intended to improve the performance of knowledgeable users, it

is important that they do not harm novice performance. Although CommandMaps remove mechanical hierarchy (i.e., hierarchy that requires physical navigation using the mouse or keyboard) as much as possible, *visual* hierarchy can still be used to provide meaningful arrangements of commands to help novice users.

4. **Maximise the workspace display area.** When using a desktop application, the user's attention is likely to be on their workspace, such as the document, spreadsheet, or image they are editing. Commands must be available on demand, but when unused, they produce visual clutter and consume space that might be better reserved for the workspace. CommandMaps maximise the workspace by using a modal separation of workspace and commands.

5.2 Performance modelling: CommandMaps, menus, and Ribbons

One way to analyse a new interface design, before investing effort into implementation and user studies, is to mathematically model its performance and compare the results to those of existing systems. In this way, interface designers can gain information about the relative merits of the new interface and decide whether it is worth implementing.

This section describes how CommandMaps were compared to menus and Ribbons using the theoretical Search, Decision, and Pointing (SDP) model [1, 33]. SDP was specifically designed to model performance with menu systems across hierarchical structures and levels of expertise. The use of SDP in this section also accounts for the proportion of selections requiring the previously selected parent item to be changed.

The SDP model [1, 33] calculates the time to select an item as the sum of time taken at each hierarchical level. The time taken at each level is the key component of the model, and is calculated as the "search/decision time" plus the pointing time (from Fitts' Law). Search/decision time depends on whether the user can decide about an item's location or must visually search for it, with experts being able to make spatial decisions, while novices must rely on visual search. Decision time uses the Hick-Hyman Law of choice reaction time [86, 96], which is a logarithmic function of the number of equally probable choices. Visual search time is a linear

function of the number of candidates. The transition from novice visual search to expert decision is modelled using a power law of practice [152]. The reader should refer to Cockburn et al. [33] and Ahlström et al. [1] for a more detailed explanation of the SDP modelling process.

5.2.1 Model assumptions and theoretical performance issues

Using the model to compare CommandMaps, menus, and Ribbons exposes several important theoretical issues about their use. In particular, the modelling process demonstrates that knowledgeable use of CommandMaps involves a single decision and pointing activity, while menu use involves two (one for selecting the right menu, and another for selecting the item). Ribbon use is more involved, depending on whether the Ribbon is minimized or not and on whether the target item is within the current tab (details below).

To simplify modelling, a series of assumptions are made. First, the model assumes a set of 210 commands that are evenly divided across seven groupings (approximately reflective of Microsoft Word), with selection of each command being equally probable. Second, command selections are assumed to begin with the cursor located at the centre of the workspace. Third, tab/menu targets are assumed to be 20 pixels wide, and Ribbon items 40 pixels wide. Error-free performance is also assumed. The predictions presented below were calculated in a simple spreadsheet using previously published calibration parameters [33].

Performance predictions are made for both novice and expert users. Novice predictions are displayed in Figure 5.4, and expert predictions in Figure 5.5. Further assumptions are presented below about the cognitive and physical processes involved in interacting with each interface.

CommandMap assumptions

Novice selections are modelled as requiring a two level search process (i.e., the hierarchical search process described by Hornof [92]): first searching for the appropriate tab marker in the CommandMap, then searching for the desired command within that group. While two levels of searching are required, only a single pointing activity is necessary in the flat display. Experts are modelled using a single-level decision between all commands, followed by a single pointing ac-

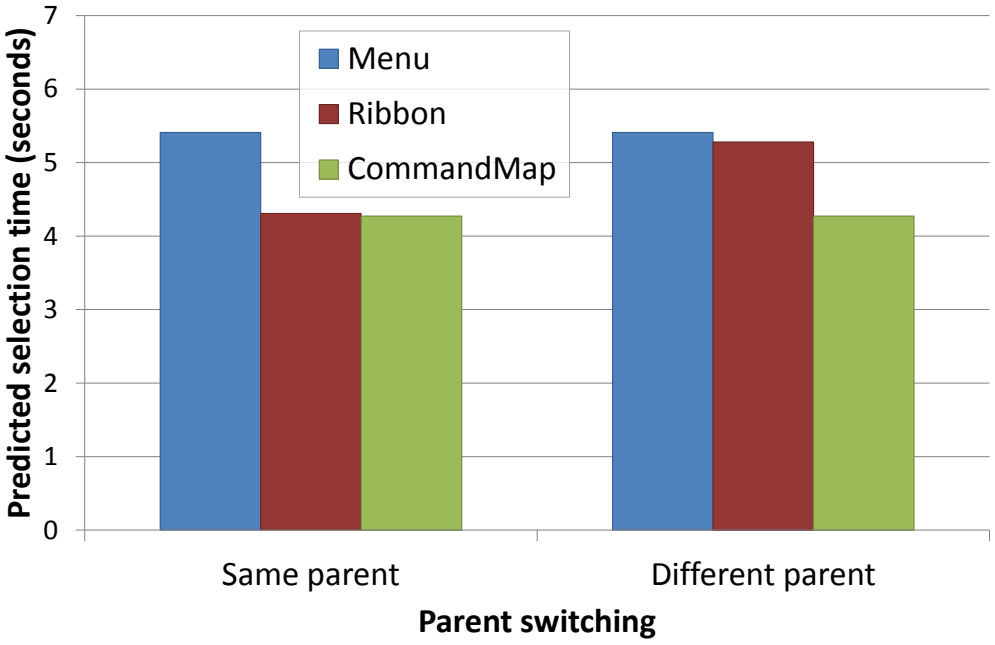


Figure 5.4: Predictions of novice selection time with menus, CommandMaps, and two alternative Ribbon models.

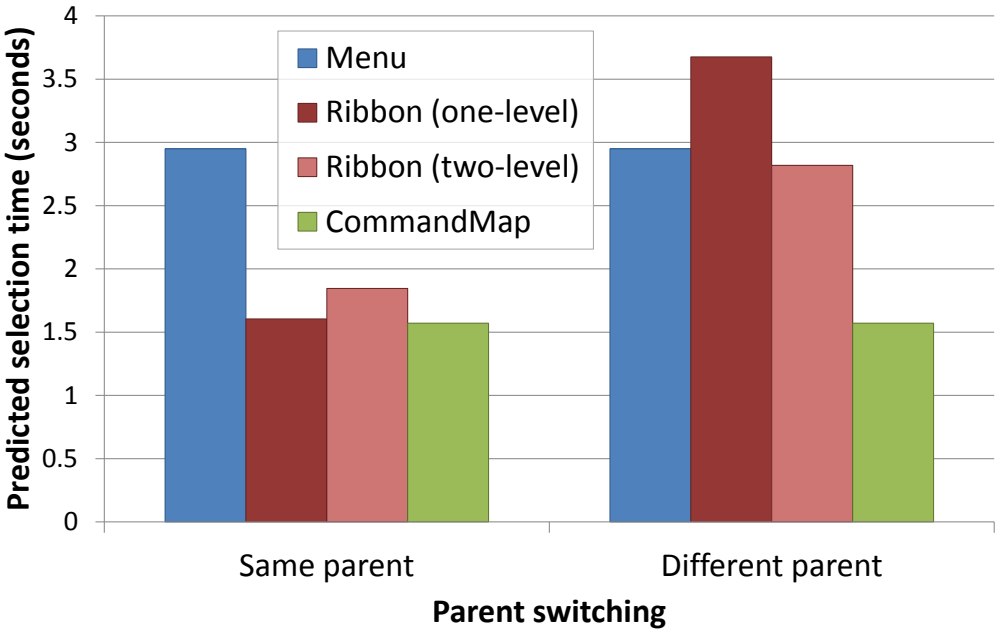


Figure 5.5: Predictions of expert selection time with menus, CommandMaps, and two alternative Ribbon models.

tivity. The mean pointing amplitude with CommandMaps is assumed to be 250 pixels.

Figure 5.5 shows expert performance predictions with the three interfaces depending on whether a switch between parent categories is required. CommandMaps are predicted to have constant fast performance of approximately 1.5s. Their speed is due to the single decision/pointing activity regardless of the need to switch from the previously selected parent category.

Menu assumptions

All selections, regardless of expertise, involve a two level acquisition process. Users first search for (novice) or decide about (expert) the menu and point to it. They then search/decide and point to the item in the menu. A mean amplitude (pointing distance) of 500 pixels from the screen centre to the top level menu is assumed, and 300 pixels for second level selections (half way through a 30 item menu).

Figure 5.5 shows a constant expert menu prediction of approximately 3s. This slow performance is due to the two decisions and pointing actions for every selection.

Ribbon assumptions

The Ribbon can be minimised, causing it to disappear after each selection, which requires a tab to be clicked before it reappears. In this case Ribbon interaction (and model) is nearly identical to menus, involving a two-level search/decision and pointing process.

Modelling performance with the non-minimised Ribbon, however, is theoretically interesting because it is sometimes necessary to switch tabs to access the desired command and sometimes unnecessary. For novices a two level searching process (as for CommandMaps and menus) is used; however, time for first-level pointing is only included when a tab-switch is necessary.

For experts, it is unclear whether acquisitions involve a single decision for a 'global' target (e.g., the user thinks "Bold" and recalls its spatial location) or two decisions (e.g., the user thinks "Home tab", "Bold"). If two decisions are involved, then selections within the currently selected tab involve a superfluous decision,

wasting a small amount of time. However, if only a single decision is made then users are likely to encounter mode errors when tab changes are required – for example, the user thinks "Bold", recalls its location from memory, and encounters a mode error when the target is not where expected because the 'View' tab is selected. Anecdotal reports suggest that Ribbon users do make frequent mode errors, lending support to the one-level decision model.

Figure 5.5 shows expert predictions for both one- and two-level Ribbon models (using the same pointing distances as menus). Ribbons are predicted to match CommandMaps when the desired tab is already visible, but to be substantially worse when a tab switch is required. Note that the one-level model predicts that Ribbons will also be worse than menus when selections involve a tab switch (due to mode error).

5.3 Experiment 5A: Expert CommandMap performance

The performance models presented in the previous section suggest that for experts, CommandMaps should be significantly more efficient than menus, and more efficient than Ribbons when tab switching is required. This sections presents an experiment designed to empirically validate this model.

The experiment described here studies simulated expert interaction, where participants are primed with knowledge of item locations. Novice performance is studied later, in Section 5.4. A third experiment, studying different design solutions to variable display space, is presented in Section 5.5. Participants completed the three studies back-to-back in a single one-hour session.

5.3.1 Hypotheses

The primary aim of CommandMaps is to improve performance by knowledgeable users who have developed spatial awareness of command locations. This experiment therefore examines the following hypotheses:

H1: Knowledgeable users can select commands faster using CommandMaps than when using Ribbons and menus.

H2: There is no performance difference between CommandMaps and Ribbons when selecting commands contained in the most recently used tab, but Com-

mandMaps are faster than the Ribbon for tasks requiring switching between different parent tabs.

H3: Subjectively, users will prefer CommandMaps.

Hypotheses 1 and 3 are important but straightforward performance and preference comparisons. Hypothesis 2 is more nuanced, examining the theoretical performance model's assumptions. As the one-level model of Figure 5.5 shows, no difference is predicted between CommandMaps and Ribbons for non-switching tasks. However, the model also predicts that CommandMaps will perform much better than Ribbons and menus when switching is required.

5.3.2 Procedure

To achieve the interface familiarity necessary to examine knowledgeable user performance, the experiment was based on a widely used desktop application: Microsoft Word 2007. All participants completed tasks using three interfaces: a Ribbon replicating the actual Word Ribbon, a menu, and a CommandMap. The menu design used seven top-level menus matching the Ribbon's tabs, with underlying menus containing all of the items in each tab, and similar group separation (Figure 5.6). The CommandMap, shown in Figure 5.2, presented all of the Ribbon tabs laid out from top to bottom within the window. None of the interfaces implemented third level pop-up/drop-down items – for example, clicking on the colour swatch drop-down arrow did not post the associated dialog.

As participants may not have encountered the Word commands used in the experiment, and because no participant could have had prior experience with the tailor-made menu or CommandMap interfaces, they were required to complete two blocks of tasks with each interface: familiarisation and performance. The familiarisation block was used to assure familiarity with the location of commands in each interface condition, while the performance block was used for experimental analysis.

Tasks were initiated by clicking a 'Click to begin next trial' button in the centre of the window (Figure 5.7), which displayed a sidebar prompt containing the name and icon for a target. Task timing began when the prompt was displayed and ran until the correct item was selected. Incorrect selections produced an audible

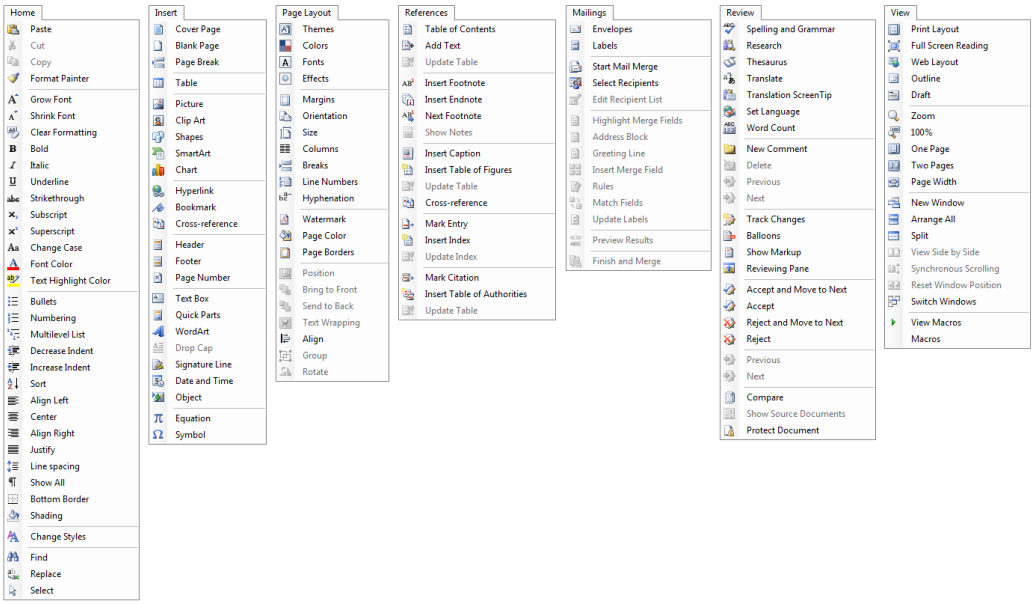


Figure 5.6: The seven menus used in Experiment 5A. To facilitate comparison, each menu was designed to be analogous to one of the tabs in the Ribbon.

beep. Participants were instructed to complete tasks "as quickly and accurately as possible".

Three sets of command targets were generated, with each set consisting of a total of six commands located in three different tabs: three in the Home tab, two in the Insert tab, and one in the View tab. Each participant used the same command set for familiarisation and performance with one interface, and then different command sets for subsequent interfaces. The order of command set and interface was counterbalanced using a Latin square.

The familiarisation block comprised 30 trials, with 5 selections for each of the six targets. The performance block contained 90 trials, with 15 selections for each of the same six targets. The order of target selection within each condition was established with a one-off random process, where the selection sequence was repeatedly regenerated until it met the constraint that 50% of selections would involve a tab switch when using the Ribbon.

Participants completed NASA-TLX [80] worksheets after each interface, and at the end of the experiment they ranked the three interfaces for preference.

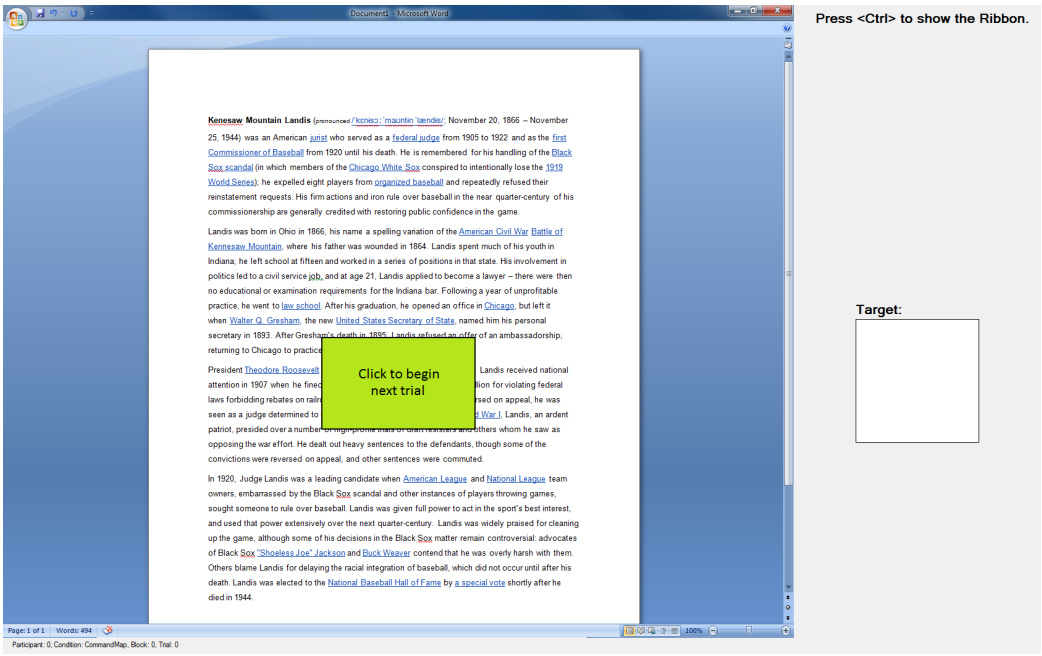


Figure 5.7: The experimental system used in Experiment 5A (CommandMap condition). The name and icon of each target command was displayed in the 'Target' box to the right.

5.3.3 Participants and apparatus

18 participants were recruited from the University of Canterbury (16 male, 2 female). The experiment was performed on a Windows 7 desktop with a 2.66 GHz Intel Core 2 Quad and 8GB of RAM. A 22 inch monitor was used, running at a resolution of 1680×1050.

5.3.4 Design

The experiment is designed as a 3×2 analysis of variance for within-subjects factors Interface (*ribbon*, *menu*, *commandmap*) and Parent (*same*, *different*). The Parent factor allows analysis of the impact of moving between different interface structures – tasks are *same* when the current selection occurs in the same menu or Ribbon tab as the last one; otherwise they are *different*. The dependent measures are task time and error rate.

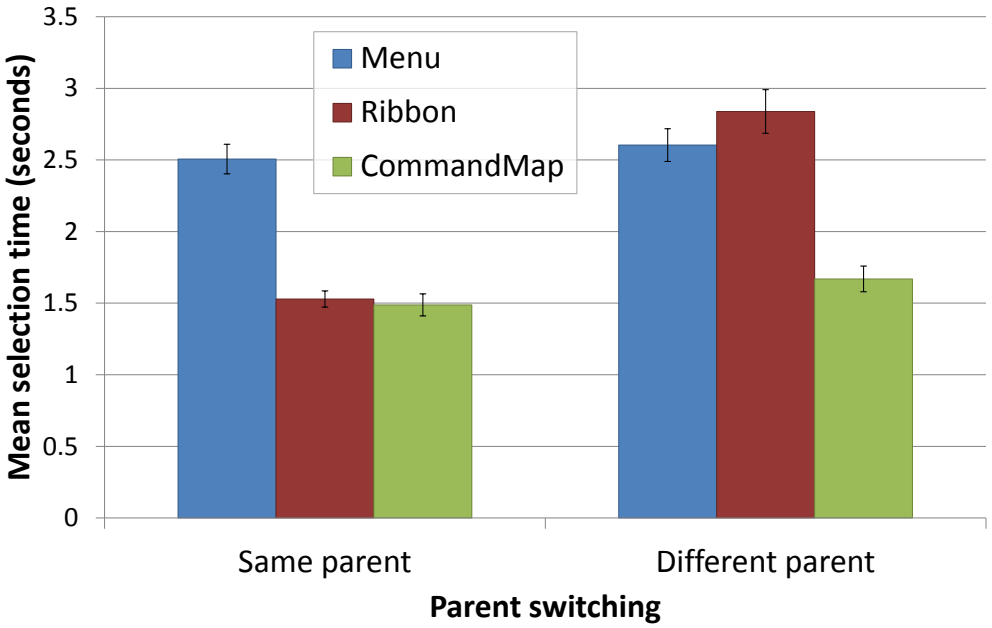


Figure 5.8: Mean selection times in Experiment 5A. Error bars show standard error.

5.3.5 Results

Task time data was analysed with and without trials containing incorrect selections, with both analyses producing the same statistical outcomes.

Mean acquisition times (errors removed) were fastest with *commandmap* (1.57 s, s.d. 0.4), followed by *ribbon* (2.11 s, s.d. 0.8) and *menu* (2.40 s, s.d. 0.4), giving a significant main effect of Interface: $F_{2,34} = 114.0, p < 0.001$. Bonferroni-corrected pairwise comparisons (total α = .05) confirm that *commandmap* was faster than *ribbon* (by 25%) and *menu* (by 34%). This provides support for **H1**.

As expected, there was a significant effect of Parent ($F_{1,17} = 155.5, p < 0.001$), with *same* parent selections faster than *different*. Importantly, though, there was a strong Interface $\times$ Parent interaction ($F_{2,34} = 187.4$, $p < 0.001$). This is shown in Figure 5.8: *commandmap* and *ribbon* performed similarly for *same* tasks, but *commandmap* was relatively faster in *different* tasks. This provides support for **H2**. The model predictions shown in Figure 5.5 are confirmed by Figure 5.8, including the crossover effect of Ribbon performance becoming worse than menus

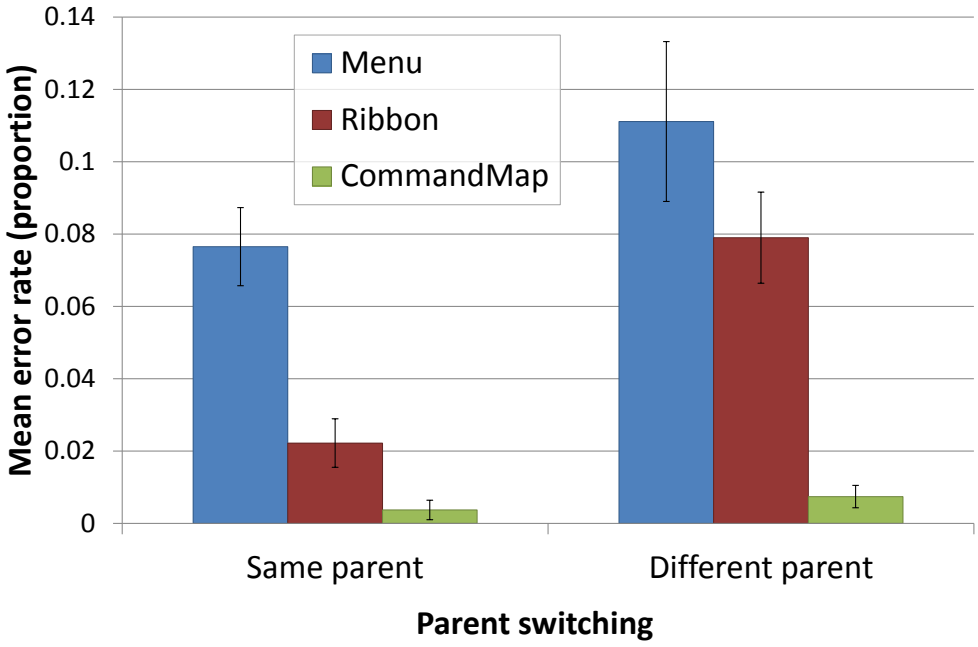


Figure 5.9: Mean error rates in Experiment 5A (as a proportion of number of trials). Error bars show standard error.

in different tasks.

The proportion of trials containing an error was much lower with *commandmap* (0.6%) than either *ribbon* (5%) or *menu* (9%): $F_{2,34} = 21.6$, $p < 0.001$. A significant Interface $\times$ Parent interaction ($F_{2,34} = 5.26$, $p < 0.05$), evident in Figure 5.9, is caused by *commandmap* error rates being relatively unaffected by *parent*, while *ribbon* and *menu* have much higher error rates in *different* parent tasks (suggestive of the hypothesised mode errors).

The combination of time and error data is important, as it shows that CommandMaps do not increase errors to achieve their improved temporal performance – they are both faster and more accurate than menus and Ribbons.

User response to CommandMaps was positive, with 14 participants ranking it as their first preferred interface, two rating Ribbons first, and two menus: $\chi^2 = 16.0$, $p < 0.001$. CommandMaps were also rated as having the lowest workload on all significant NASA-TLX measures (Table 5.1). This provides support for **H3**.

	Menu	Ribbon	CM	χ_r^2	Significance
Mental demand	3.1 (1.1)	3.4 (0.9)	2.5 (1.2)	11.9	< 0.005
Physical demand	3.7 (1.1)	3.5 (0.9)	2.4 (1.0)	11.6	< 0.005
Temporal demand	2.9 (1.1)	3.2 (0.9)	2.4 (1.2)	9.3	< 0.01
Hard work	3.1 (0.9)	3.1 (1.0)	2.0 (1.1)	10.5	< 0.01
Frustration	3.3 (1.0)	2.9 (1.0)	1.9 (1.1)	13.4	< 0.005

Table 5.1: Mean (standard deviation) NASA-TLX responses (1 = low, 5 = high).

5.4 Experiment 5B: Novice use of CommandMaps

CommandMaps are primarily intended to enhance knowledgeable users' performance, but novice performance is also important. Since CommandMaps display all commands at once, there is a risk that visual search performance will be impaired by the need to visually scan many concurrent candidates. This study therefore compares novice performance with CommandMaps, Ribbons, and menus.

5.4.1 Procedure

The experiment involved pointing to and selecting randomly arranged targets in logical groupings using CommandMap, menu, and Ribbon interfaces. Five groups of 24 items each were created to populate the interfaces: *animals*, *cartoon characters*, *food*, *office items*, and *sports* (Figure 5.10). Only items from *animals*, *food*, and *sports* were used as targets. The groups were intentionally unconnected with computing to avoid transfer effects from traditional interface experience.

Tasks were presented to participants using an identical prompting interface to Experiment 5A. Participants completed twenty-four tasks with each interface before proceeding to the next interface (interface order counterbalanced using a Latin square). The tasks with each interface comprised selecting eight unique targets in each of three different groups (e.g., eight different animals). The order of task presentation was manipulated such that half of the tasks involved switching parent group and half did not (to test the impact of searching within and across tabs). To reduce learning effects across tasks (and hence emulate novice visual search) no target item was reused throughout the experiment, and the location of all items (parents and items within groups) was randomised for every trial. Participants provided comments and rated the ease of finding targets at the conclusion

Figure 5.10: The CommandMap interface used in Experiment 5B.

of each interface condition, and at the end of the experiment they ranked the three interfaces for perceived performance and preference.

Participants, apparatus, and design are identical to Experiment 5A.

5.4.2 Results

Mean acquisition times were similar with *commandmap* (4.45 s, s.d. 1.73) and *ribbon* (4.38, s.d. 1.4), but slower with *menu* (5.74, s.d. 1.6), giving a significant main effect of Interface $F_{2,34} = 110.9$, $p < 0.001$. In pairwise posthoc comparisons (Bonferroni adjusted t-tests), *menu* was slower than both *ribbon* and *commandmap*, but there was no difference between *commandmap* and *ribbon* ($T_{17} < 1$).

There was a significant Interface×Parent interaction ($F_{2,34} = 12.3$, $p < 0.001$; Figure 5.11), with *ribbon* slightly faster than *commandmap* for *same* parent selections, but *commandmap* slightly faster than *ribbon* for *different* parent selections. Pairwise comparisons between *commandmap* and *ribbon* in each of these conditions (*same* and *different*) show no significant difference ($p > 0.05$).

Error analysis showed a 2.8% error rate with *commandmap*, 5.1% with *ribbon*,

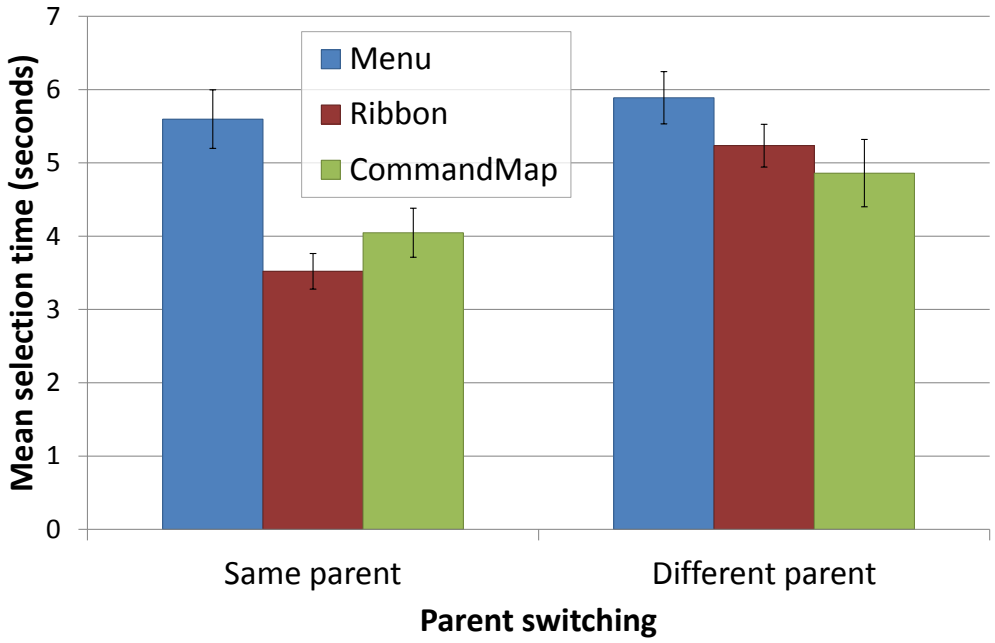


Figure 5.11: Mean selection times in Experiment 5B. Error bars show standard error.

and 16% with *menu*: $F_{2,34} = 35.2$, $p < 0.001$. There were marginally more errors with *different* parent (9.2%) than with *same* (6.6%): $F_{1,17} = 4.1$, $p = 0.06$. There was no Interface×Parent interaction $F_{1,17} < 1.0$.

Subjective responses to the question "It was easy to find targets" (1 disagree, 5 agree) indicated greatest ease with *commandmap* (mean 3.5, s.d., 1.0), followed by *ribbon* (3.2, 1.0) and *menu* (2.4, 0.9): Friedman $\chi^2 = 10.0$, $p < 0.005$. Eleven participants ranked *commandmap* as their preferred interface for the task, four preferred *ribbon*, and two preferred *menu*: $\chi^2 = 7.9$, $p < 0.05$. Comments on the CommandMap presentation were mixed, with one participant stating "Too much to see at once", and another saying "I like how you can see all the buttons at once."

The key finding is that novice performance is similar when using CommandMap and Ribbon designs; both are substantially better than menus.

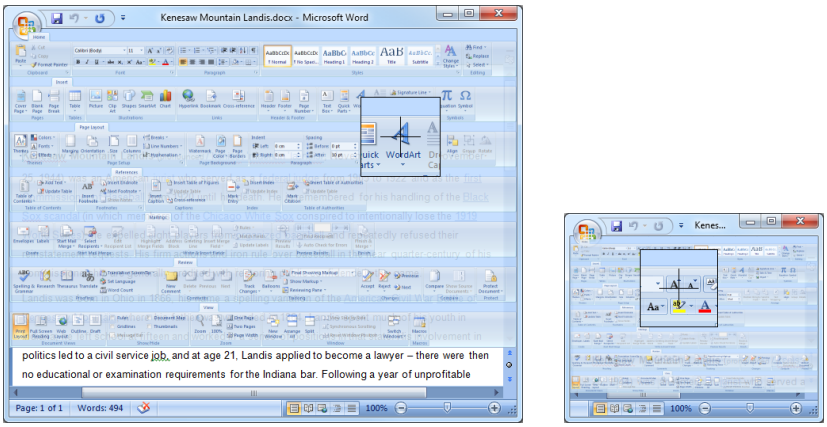


Figure 5.12: The Scaling CommandMap, with dimensions scaled to 50% (640×512 px) and 25% (320×256 px) of original size.

5.5 Experiment 5C: CommandMaps and window geometry

Experiments 5A and 5B used large, static windows, but any practical deployment needs to accommodate variable window sizes and positions. This raises questions of how CommandMaps should respond to window geometry manipulation, and how this affects their performance. The following sections describe and test two CommandMap designs for responding to window geometry manipulation – one based on scaling within the window boundary, and another using a pop-up window that extends beyond the window boundary (similar to the way a standard posted menu is not constrained to its parent window).

5.5.1 Scaling and Pointing Lens CommandMaps

Scaling CommandMaps are dynamically resized in response to window size manipulations so that items maintain relative spatial location (Figure 5.12). To avoid distortion when windows are resized on only one dimension, they maintain a 1:1 aspect ratio using the smaller window dimension. They are anchored to the top-left corner of the window. To assure that targets remain discernable at small scales a pointing lens is used to magnify the area under the cursor.

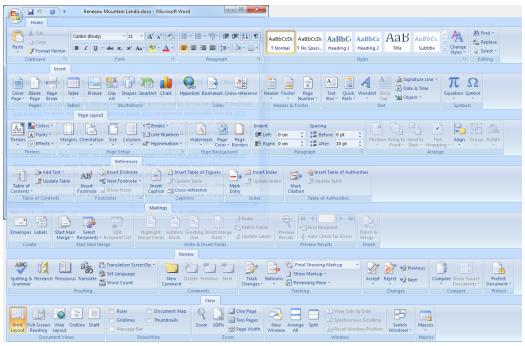


Figure 5.13: The Pop-up CommandMap, in a window with 50% width and height (640×512 px).

5.5.2 *Pop-up CommandMaps*

Pop-up CommandMaps are displayed in a pop-up window of constant (full) size. Like standard menus, the CommandMap is anchored in the top-left window corner by default, but is repositioned outside the window boundary when necessary for the entire CommandMap to appear within the display (Figure 5.13). Therefore, when the window is small, or when the window intersects a screen edge, the CommandMap extends outside the window boundary.

5.5.3 *Evaluating the designs*

The experiment compared knowledgeable user performance with scaling and pop-up CommandMaps at three different window sizes: full size (1280×1024), 50% (640×512), and 25% (320×256). The full size just showed the basic CommandMap, and was therefore identical in both designs. The 50% size represents a realistic lower bound for window size with a standard desktop application. The 25% size represents an extreme limit of interaction.

5.5.4 *Procedure*

Experimental tasks involved selecting the same six targets used for the command-map condition in Experiment 5A. Participants initially performed a block of 'refresher' trials, selecting each of the six targets twice (data discarded). They then made 36 selections with the scaling interface and 36 with the pop-up interface, with interface order counterbalanced. The 36 selections comprised 12 at each size

(full, 50% and 25%), consisting of two repetitions of each of the six targets. The targets were ordered such that each selection used a different window size to the preceding one (e.g., a participant might select target 1 at full size, then target 2 at 25%, then target 3 at 50%, and so on) in order to maximise abrupt transitions between window sizes. Tasks were presented to users using the same prompting interface as Experiments 5A and 5B.

5.5.5 Participants, apparatus, and design

Participants and apparatus are identical to Experiments 5A and 5B. The design is a 2×3 RM-ANOVA for within-subjects factors Interface (*scaling*, *popup*) and Size (*full*, *50*, *25*). The main dependent measure is task time.

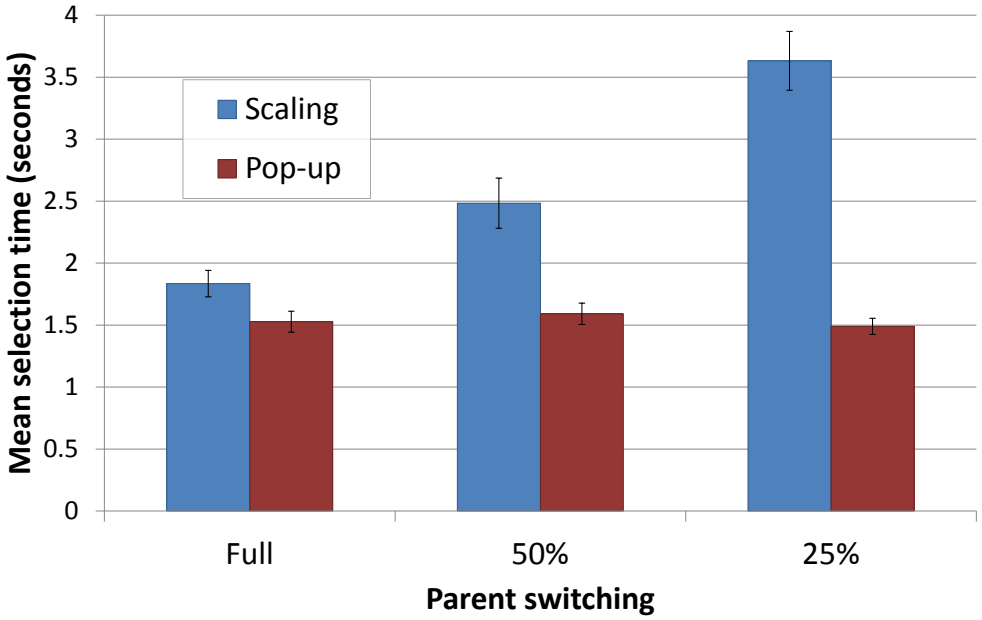


Figure 5.14: Mean selection times in Experiment 5C. Error bars show standard error.

5.5.6 Results

The error rate was low (a total of 10 across 1296 trials), so error analysis was not conducted. *Popup* (mean 1.54 s, s.d. 0.33) was much faster than *scaling* (mean

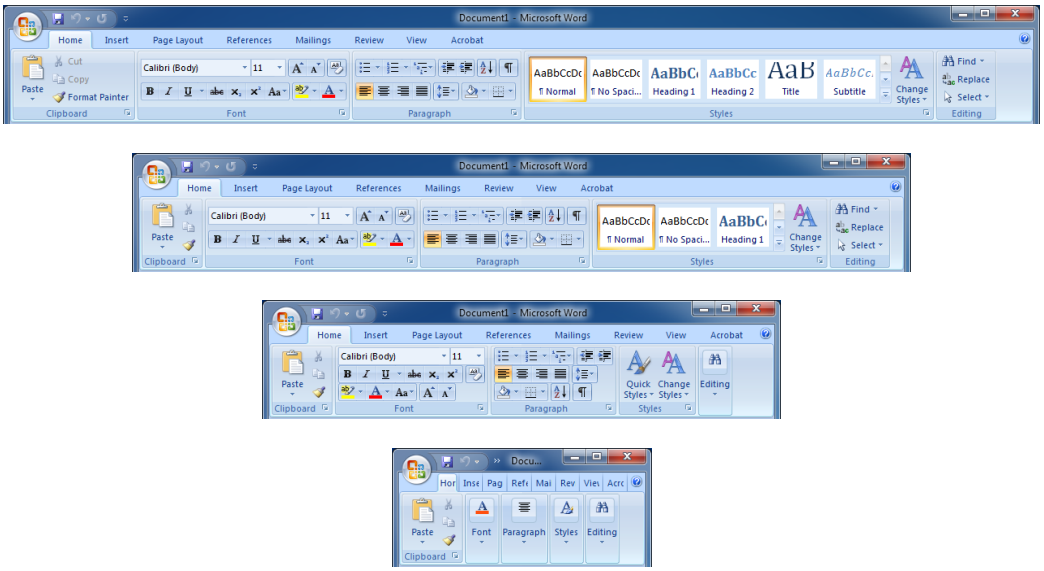


Figure 5.15: The Word Ribbon at widths of 1280px, 960px, 640px and 320px. As the Ribbon shrinks, additional hierarchical traversal is required to reach targets.

2.65 s, s.d. 1.1), giving a significant effect of Interface ($F_{1,17} = 82.1$, $p < 0.001$; Figure 5.14). There was also a significant main effect of Size ($F_{2,34} = 81.5$, $p < 0.001$), but this was due to *scaling* performance deteriorating as size decreased, while *popup*'s performance remained stable, leading to a significant Interface $\times$ Size interaction ($F_{2,34} = 77.7$, $p < 0.001$). *Popup* outperformed *scaling* even at *full* size, where the two conditions were identical. This suggests that the abrupt transitions between sizes were a significant detriment to performance with *scaling* – one participant commented "I found I lost my sense of where things were as the scale changed." All participants preferred the pop-up interface.

Pop-up's performance stability across window size is important. In Experiment 5A, the CommandMap mean selection time of 1.57 s was 25% faster than that of the Ribbon, and the mean selection time for Experiment 5C's pop-up condition was nearly identical at 1.54 s. A Ribbon condition was not included in Experiment 5C, but it would clearly have performed worse than it did in Experiment 5A due to its progressive elision of items into additional hierarchical levels (see Figure 5.15). The results of Experiment 5C therefore suggest that the advantage for pop-up CommandMaps over the Ribbon would exceed 25% with small windows.

5.6 Discussion

To summarise the results, Experiment 5A demonstrated that CommandMaps provide substantial performance benefits for expert users – they were 34% faster than menus and 25% faster than the Ribbon. The results confirmed the predictive performance model (Figure 5.5), including a cross-over effect with Ribbon performance being worse than menus for selections involving a parent category switch (suggesting that the one-level model of Ribbon interaction is closer to the truth). CommandMaps were also much less error prone, with 0.6% errors compared to 5% and 9% with Ribbons and menus respectively.

Experiment 5B showed that novice visual search for randomly located items in CommandMaps is faster than menus, but not significantly different to Ribbons. The study also showed that the relative performance of CommandMaps and Ribbons depends on whether selections involve switching from the previous parent category, with CommandMaps performing better when a switch is required, and Ribbons performing better when a switch is not required. Although these differences were not significant, the results diverge somewhat from the selection times predicted by the novice performance model (Figure 5.4, which underestimated the efficiency of the Ribbon for same-parent selections.

Experiment 5C demonstrated that popup CommandMaps remain efficient regardless of window size, that scaling CommandMaps performed worse the greater the scale factor, and that the disorientation caused by the scaling caused users to perform worse even when locating items in the full-size window.

The rest of this section compares the predicted and empirical results, discusses issues that were not covered by the experiments regarding real-world CommandMap use, and identifies further research on scalable interfaces.

5.6.1 Predicted vs. empirical results

The empirical results of Experiment 5A closely matched the theoretical predictions generated by the expert performance model (Figures 5.5 and 5.8). Furthermore, the preferred 'one-level model' of Ribbon use anticipated frequent mode errors when parent switches are required, as observed with the Ribbon's 5% error rate (as compared to 0.6% with CommandMaps) and the fact that the Ribbon was even slower than menus in such cases (Figure 5.8).

The theoretical model mechanically implements predictions using previously reported parameters [33] (eliminating any chance of calibration 'bias'), and the model's formulae for expert performance attend only to the number of interface levels, the timing associated with location decisions at each level, and pointing requirements. Therefore, the success of the CommandMap for expert users is attributable to its two defining properties – stability of item location (allowing spatial decisions), and maximally flattened hierarchy (allowing acquisition with a single decision and pointing action).

The empirical results of Experiment 5B, however, did not match their respective predictions as closely (Figure 5.4 vs. Figure 5.11). In particular, the model predicted Ribbon and CommandMap selection time to be approximately equivalent for same parent selections, while empirical results showed that the Ribbon actually outperformed the CommandMap for such selections. There are a number of possible reasons for this. First, the time required for visual search (which novices must employ to find items) is more variable than the time required for spatial retrieval, and this variance is not accounted for in the model. The higher variance in novice behaviour can be visualised by comparing the error bars in Figure 5.8 with those in Figure 5.11. Second, visual search in CommandMaps was modelled as a hierarchical process, as discussed in Section 3.1.1; however, it is possible that category names were not sufficiently salient or meaningful to be utilised by the participants. The higher relative selection times for CommandMaps could therefore be caused by participants performing a visual scan across the entire CommandMap. Finally, the icons used in the experiment were distinctive enough that they may have allowed participants to find items by visual pop-out (Section 3.1.1), rather than a linear scan. These effects, in combination, generate enough uncertainty to account for the difference between the predicted and empirical data.

5.6.2 CommandMaps in the real world

All three experiments focused on command selection performance, with tasks involving repeated selection of a small set of serially presented targets. While real work sometimes involves executing a series of commands (e.g., changing the zoom level, inserting a symbol, and formatting it) it normally interleaves activities

on the workspace with command selections. This raises concerns about whether the experimental findings will generalise to real use, discussed below.

Impact of the small target set on spatial memory

Experiment 5A involved repeated selections of six target items. The small set was used to ensure that participants had a good spatial knowledge of target locations (emulating expertise), but it is possible that the method induced spatial location memory that is artificially refined. However, prior work (e.g., [39, 73, 169]) has shown that people are able to retain long-term memory of large item sets, suggesting that CommandMaps would still perform well in situations with larger command sets.

Activating control

The experimental interface used the `Ctrl` key to activate the CommandMap, but this requires bimanual operation with one hand on the key and another on the mouse. Participants in the three experiments were required to issue an intense series of command selections, so it was natural for them to keep one hand on or near the `Ctrl` key. However, during real work the non-dominant hand might be otherwise occupied, demanding a homing action to the activating key. Two solutions to this concern are first, the CommandMap could be posted by clicking in a designated area (e.g., window title); similar to how the current Ribbon can be posted once 'minimized'; second, a dedicated mouse button could be used to activate the CommandMap mode, allowing unimanual selection. Similarly, on a touchscreen device, the CommandMap could be activated with a specific gesture (e.g., four finger touch).

Workspace overlay

To display the full set of commands simultaneously, the CommandMap covers the user's work or content area with a configurable transparent overlay. While this overlay allows the underlying area to remain visible, it is possible users may respond less favourably to having their content somewhat obscured when invoking commands that allow previews prior to final selection (e.g., font size). There is

also the risk that the abrupt changes between workspace and command modes may distract users and interrupt their taskflow – further study on this is needed.

Initial user reaction

Experiment 5B shows that novice visual search performance is similar between Ribbons and CommandMaps. However, there are two concerns regarding the reactions of first-time users. First, three participants indicated that the number of controls was 'overwhelming' when first viewing the CommandMap (although this impression quickly dissipated on use). Second, there is an absence of control affordance due to the omission of obvious controls at their familiar location. Both of these concerns are short-term effects that might be eased with a help display after installation.

Size of command set

While CommandMaps utilise screen real estate to a much higher degree than conventional techniques, there is still a limit to the number of commands that can be displayed at once. In situations where the available command set is too large, a hierarchical structure must still be employed. However, CommandMaps should still offer a performance increase over contemporary interfaces if the hierarchy is as shallow as possible. Furthermore, CommandMaps in their current form are unable to support certain features of the Ribbon, such as contextual tabs, due to a lack of screen space. Anyone designing a practical implementation of CommandMaps will therefore have to keep screen size limitations in mind when choosing control arrangements.

5.6.3 Implications of Experiment 5C

Experiment 5C showed that visually scaling an interface was detrimental to performance. However, the scale factors examined by Experiment 5C were extreme, and likely not representative of typical window sizes. Further research is therefore needed on the effects of scaling at more realistic sizes. This is particularly important in cases where the display itself is not large enough to contain a full-size CommandMap, such as on netbooks; in these cases, the pop-up CommandMap is

not an option, and the interface must either be scaled, or controls elided and rearranged in a manner similar to the Ribbon (Figure 5.15). Experiments examining interface scaling and other geometric transformations in-depth are presented in Chapter 8.

5.7 Conclusions

In modern user interfaces, hierarchical command organisations are common. However, users can remember the spatial locations of controls without the need for hierarchy, implying that hierarchy traversal is inefficient for experienced users.

This chapter presented the notion of combining spatial memory and flat hierarchies to support efficient command access and instantiated these ideas within CommandMaps. Performance models supporting the CommandMap design were generated, and empirically validated through two studies: one demonstrating a speed increase for expert users of 34% over menus and 25% over Microsoft's Ribbon, and the other showing no significant performance difference for novices. Subjective responses indicated that CommandMaps was preferred across both experiments. Finally, two alternative designs allowing CommandMaps to remain effective at smaller window geometries were evaluated, with a "pop-up" design performing significantly better than one that scaled widgets according to the window dimensions.

The work presented here forms the basis for future chapters. Chapter 6 studies the effectiveness of CommandMaps in more realistic tasks. Chapter 7 examines the StencilMap, an extension to the CommandMap aimed at easing visual search for novice users. Chapter 8 expands on Experiment 5C, examining in more depth the effects of geometric transformations on interfaces.

Chapter 6

Evaluating CommandMaps in Realistic Tasks

Chapter 5 presented the CommandMap design, and demonstrated through the use of modelling and command-selection experiments that CommandMaps offer faster interaction than standard interfaces such as menus and Ribbons. However, as discussed in Section 5.6, there are several issues not covered by the studies in Chapter 5 which have the potential to reduce the effectiveness of CommandMaps in real-world situations.

Therefore, the fundamental question that this chapter attempts to address is whether or not CommandMaps work in the real world. From a methodological perspective this is a critical question to answer, especially when proposing an interface mechanism that changes the basic method for presenting and accessing commands. The importance of addressing this question is accentuated by the many previous HCI studies that have demonstrated divergence between results obtained in the lab and in the field, with lab studies typically generating stronger, more positive findings [26, 45, 85, 153]. Further, the importance of triangulating results through complementary methods is well known in the behavioral and social sciences [141], including HCI [133].

More specifically, this chapter comprises an attempt to answer the following research questions:

Q1) Do CommandMaps demonstrate performance and subjective advantages in realistic tasks? Real tasks, such as document formatting or image editing, have many characteristics that were not present in previous studies. For example, an individual command selection usually forms a small step of a larger task, which may also include content creation activities such as typing or drawing. If CommandMaps somehow interfere with the other cognitive and physical activities involved in task completion, then these drawbacks may outweigh the benefits.

Q2) How do users respond when first using CommandMaps? The studies in Chapter 5 showed that the performance of users without item location knowledge was not adversely affected by CommandMaps, but participants were allowed time to become familiar with the interface first. Therefore, the issue of users' initial 'out of the box' experience, both in terms of performance and subjective impressions, requires further examination.

Q3) How do performance and subjective impressions change in real tasks as users gain experience? While the studies in Chapter 5 showed that performance and subjective preferences favour CommandMaps after condensed periods of command selection, the progression from novice to expert over a realistic period of time (e.g., a week) has not been studied.

Q4) Do CommandMaps generalise to multiple application contexts? In order to recommend the use of CommandMaps in the real world, more information is required regarding whether they are successful in different application genres (such as word processing and image editing), and whether there are characteristics of different applications that make CommandMaps more or less useful.

Q5) What usability issues arise with CommandMaps in realistic use? By observing realistic usage scenarios, usability issues can be identified and the CommandMap design can be improved.

In order to answer these questions, this chapter examines the use of CommandMaps in more realistic tasks. First, two implementations are presented that modify real-world software systems to use CommandMaps, based on Microsoft Word and the open-source painting and photo editing application 'Pinta'. These implementations serve as experimental platforms in two studies, which build on the studies described in Chapter 5 by adding elements of real-world interaction and examining users' subjective responses in more depth.

The first study, addressing **Q1**, examines how users complete moderately realistic word processing tasks when using the Microsoft Word CommandMap. For baseline comparison, an equivalent task with an unaltered version of Microsoft Word 2007 was also examined. The second study, addressing **Q2-5**, focuses on user opinions, and the evolution of those opinions, during the first week of interaction with CommandMaps. Participants interacted with a CommandMap and the standard interface in both Microsoft Word and Pinta. They were given realistic tasks to complete; they were invited to complete them in any way they saw

fit; and they repeated the tasks every day for a week. Data was collected from observations of use, interviews, and questionnaires throughout the week.

6.1 Real-world CommandMap implementations

In order to evaluate CommandMaps in realistic tasks, a functional CommandMap implementation was required. To assess the feasibility of CommandMaps in multiple application domains, two CommandMap systems were developed: one for Microsoft Word, and one for Pinta, an open-source image editing application.

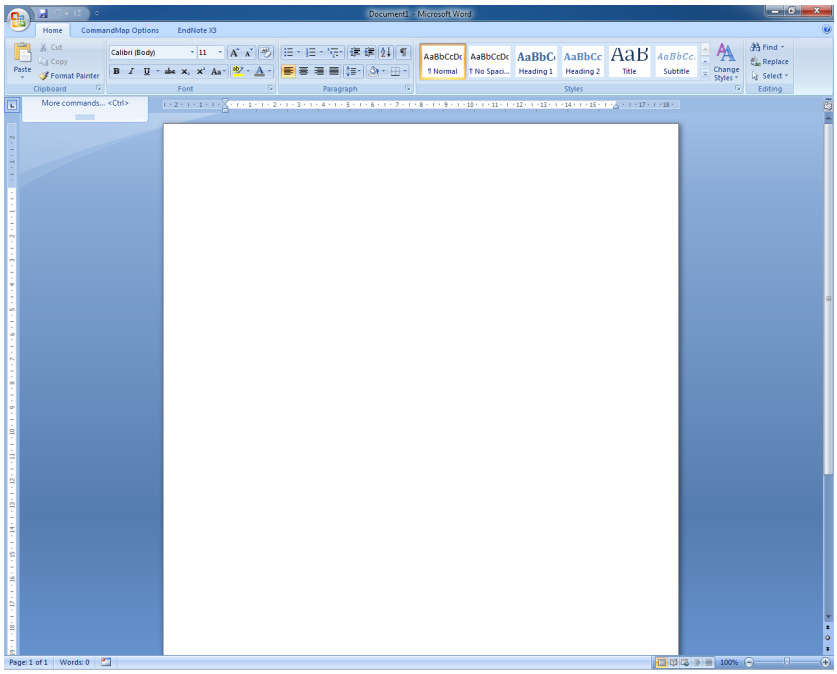
6.1.1 Developing a CommandMap for Microsoft Word

The implementation of the Microsoft Word CommandMap is based on the prototype described in Chapter 5, with some adjustments (Figure 6.1). The original prototype displayed no commands until a command key was pressed, at which point all of the commands were displayed in parallel, overlaying the document. This design was modified slightly so that the Home tab was continually displayed at the top of the window (see Figure 6.1a). There were two main reasons for this: first, it provides convenient visual access to the most frequently used commands; and second, it provides a visual focus of attention for displaying a tip for novice CommandMap users that instructs them on how to access other commands. This tip takes the form of a dropdown tab with the label "More commands... `Ctrl`" (see top-left of Figure 6.1a). When the "More commands" tab is clicked, or when the `Ctrl` key is held, the CommandMap is displayed, making all commands from all tabs available at once by pointing and clicking (Figure 6.1b). Releasing the `Ctrl` key without making a selection cancels the command and hides the CommandMap. Selecting a command completes the command and hides the CommandMap.

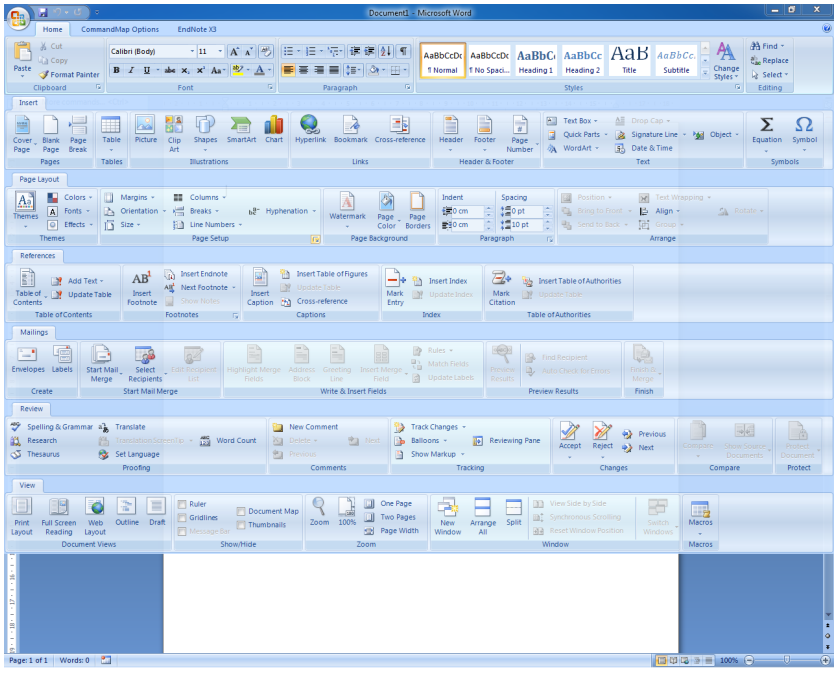
Implementation and design

The CommandMap is implemented as a plug-in for Microsoft Word 2007. The plug-in uses an open-source, third-party Ribbon control[1] to mimic the look and

[1] http://officeribbon.codeplex.com/



(a) The default state of CM-Word.



(b) CM-Word with CommandMap visible.

Figure 6.1: The CommandMap version of Word.

feel of the original Word 2007 Ribbon. The visual appearance of controls is replicated using the original Word 2007 icons. When the CommandMap is displayed, either by holding `Ctrl` or by clicking the on-screen drop-down button, the CommandMap appears as a borderless overlay at 95% opacity on top of the main Word window. It displays tabs in a vertical arrangement below the already-visible Home tab. Interaction with the controls in the CommandMap works the same way as the Ribbon: clicking an item invokes its corresponding command using the Word Object Model.

Although the plug-in provides good coverage of Microsoft Word's interface capabilities in a fluid and convincing user interface, certain features of Word were difficult or impossible to replicate using the plug-in approach. For example, contextual tabs, which are only visible in certain editing contexts (such as when a table or image is selected), still appear in the topmost Ribbon, as do the tabs of other Word plug-ins, such as EndNote. "Galleries" of pre-defined content (such as the *WordArt*, *Symbol* and *Equation* menus) were also excluded, due to the significant amount of engineering required to replicate them.

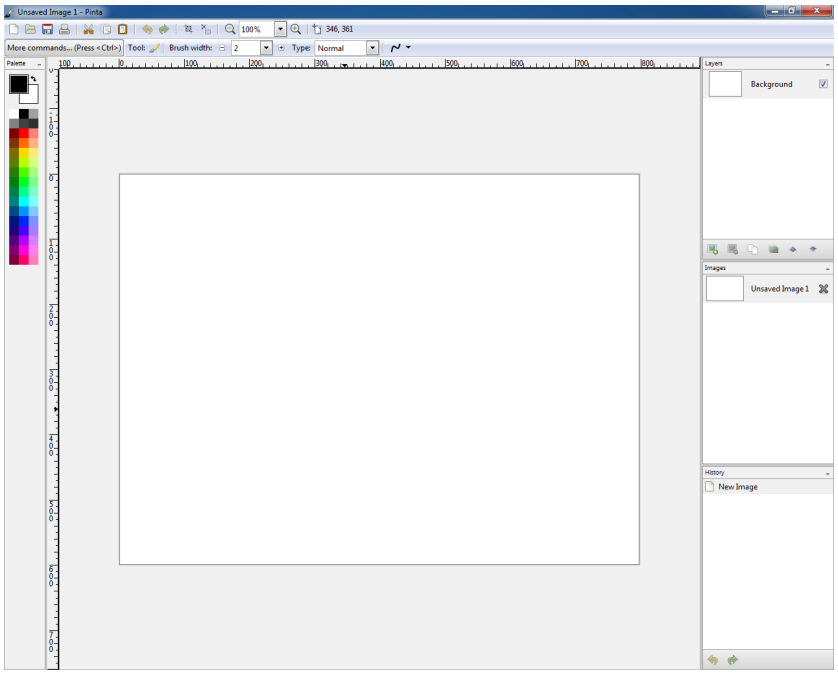
6.1.2 Developing a CommandMap for Pinta

In order to explore the application and use of CommandMaps in a domain other than word processing, a CommandMap was also designed and implemented for an image editing application. The image editing domain was chosen because content creation and editing of images is predominantly mouse-based, which is substantially different to the keyboard-based creation and editing actions involved with word processing. Pinta[2], a fork of Paint.NET, was selected as the base application because its C♯ codebase is open source. The adapted version of Pinta replaces its menus with a CommandMap, requiring various interface adaptations to the CommandMap concept, as described below.

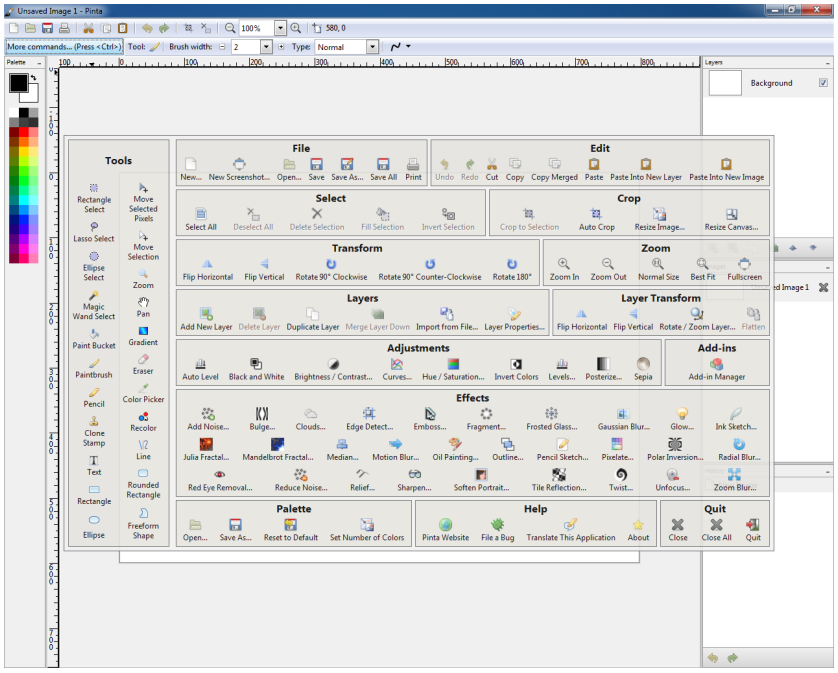
6.1.3 Implementation and design

Pinta is implemented in C♯, using the .NET Framework. The original code was forked and modified to create a CommandMap-based version, as shown in Figure 6.2. Image editing programs such as Pinta typically have toolbars, as well

[2] http://pinta-project.com/



(a) The default state of CM-Pinta.



(b) CM-Pinta with CommandMap visible.

Figure 6.2: The CommandMap version of Pinta.

as various palettes (such as *Tools*, *Colors*, *Layers*, and *History*) that are docked around the edge of the screen. This raises the design question: when converting an application's interface to a CommandMap, what subset of controls should the CommandMap contain?

User interface toolbars and palettes are often used to provide convenient ready-to-hand access to related controls, thus obviating the need for the user to navigate through menu structures for those controls. These toolbars and palettes, therefore, serve a related function to CommandMaps, although for much smaller command subsets. In the modified version of Pinta, the existing always-visible palettes and toolbars were left as-is, and only the menu-based commands were included in the CommandMap. One exception was the Tools palette, which was moved into the CommandMap to encourage study participants to use the CommandMap more often.

6.2 Experiment 6A: Word processing

The goal of Experiment 6A was to answer **Q1) Do CommandMaps demonstrate performance and subjective advantages in realistic tasks?** The experiment therefore compared user performance and subjective assessments for a word processing task when executed using the unmodified version of Microsoft Word, and when using it with the CommandMap plug-in installed. Participants were given written instructions describing the formatting of a document. The tasks involved elements of text selection, command execution and typing. Each participant completed the tasks using both versions of Microsoft Word.

6.2.1 Participants and apparatus

Ten participants (7 male, 3 female; aged 18-34 years) were recruited from the University of Saskatchewan. All were familiar with Microsoft Word 2007, using it for a mean of 16.1 hours/week (std. dev. 9.6), and none had seen a CommandMap UI before.

The experiment was performed on a quad-core Windows 7 machine with Microsoft Word 2007 installed. Participants used a 23 inch monitor running at 1920×1080 resolution, and a standard keyboard and mouse.

6.2.2 *Procedure*

Participants first completed a demographics questionnaire. They then completed two experimental tasks ('Task One' and 'Task Two', see Table 6.1), with one performed using the CommandMap and the other using the standard Word Ribbon. The order of CommandMap and standard Ribbon interfaces was counterbalanced between participants. To familiarise the participant with both the upcoming interface and the commands used in the upcoming task, a brief practice task (not timed) was performed immediately prior to both Task One and Task Two. Participants timed themselves by clicking on-screen Start and Stop buttons before and after each task, and were instructed to complete tasks as quickly and accurately as possible.

In order to measure initial impressions of the CommandMap and analyze whether opinions changed after actual use, a Likert-scale questionnaire (Table 6.3) was administered immediately after participants were first shown the CommandMap, and an equivalent one was given after participants had completed a task with it. Participants also completed a NASA-TLX [80] questionnaire after using each interface (Table 6.2) and gave written comments.

6.2.3 *Tasks*

With each interface, participants performed a practice task followed by a timed task. The nature of the task and the instructions were nearly identical between the practice task and the following experimental task, with the only difference being that the practice task was performed on a smaller document requiring fewer changes. Tasks One and Two were always performed in the order One, then Two (although, to repeat, the interface order was counterbalanced between them). Table 6.1 gives a summary of the operations required in the two tasks.

To minimise unintended learning effects, the tasks were designed to be isomorphic, but not identical (Word Count was the only command used in both). As Experiment 6A is primarily concerned with performance issues and initial impressions, the experimental tasks are designed to focus on the key difference between the user interfaces – their methods of command selection. It is worth noting, however, that command-heavy usage is not always unrealistic: for example, Lafreniere et al.'s study of application use [117] showed single sessions that contained up to

Task One operations	**Task Two operations**
Select text	Select text
Click *Accept Change* (automatically moves to next change)	Click *Reject Change* (automatically moves to next change)
Go back and re-select accepted text	Go back and re-select rejected text
Click *Strikethrough*	Click *Superscript*
25 repetitions of above	
Select paragraph	Select paragraph
Click *Word Count*	Click *Word Count*
Memorise number of words	Memorise number of words
Close *Word Count* dialog	Close *Word Count* dialog
Click *Insert Footnote*	Click *Insert Comment*
Type word count into footnote	Type word count into comment
14 repetitions of above	

Table 6.1: The operations required for each task in Experiment 6A.

1489 separate command invocations. Experiment 6B, reported later, examines less constrained experimental tasks.

6.2.4 Design

The experimental analysis was structured as a 2×2 mixed design ANOVA, with within-subjects factor Interface (*standard*, *commandmap*) and between-subjects factor Task Pairing (*cm-first*, *standard-first*). Task Pairing was included as a factor in order to check for any asymmetric skill transfer effects, which might stem from one of the interfaces in Task One serving as better preparation for the second interface in Task Two. The primary dependent variable was task completion time; TLX responses for each condition were also analysed, as well as responses to the Likert-scale questionnaires before and after use of the CommandMap.

6.2.5 Hypotheses

Previous studies of CommandMaps examined command selection time in response to a stimulus that identified the target command 5.7. In this experiment, the stimuli for command selections are placed into a more realistic interaction context, where selections are made to enact actual changes to the document. This difference could potentially influence experimental results for CommandMaps, for at least two reasons. First, in making a series of changes to a document, partici-

pants are likely to be expending some of their cognitive resources on activities such as remembering their location within the document – this may decrease the advantage of using a CommandMap, which relies on spatial memory. Second, using the CommandMap involves a substantial display context switch, with the CommandMap temporarily obscuring the work surface. This may interrupt the user's higher-level task flow, including their memory of their location in the document, how far through the task they were, or what they were supposed to do next. However, the same risks apply to standard user interfaces – for example, a user searching through tabs in the Ribbon to find a command may be just as likely to lose task flow.

The hypotheses for this study are therefore that the previously observed benefits for CommandMaps would continue to hold. Formally, they are:

H1: Subjects will complete tasks more quickly with the CommandMap than with the standard interface.

H2: Subjects will prefer CommandMaps to the standard interface.

In addition, an exploratory comparison was planned between the Likert-scale questionnaire measures before and after CommandMap use (Table 6.3), in order to quantify whether users' initial impressions of CommandMaps changed.

6.2.6 Results

Task time

A two-way ANOVA on task time showed a significant main effect of Interface ($F_{1,8} = 31.6$, $p < 0.001$, $\eta^2 = 0.8$), with *commandmap* (mean 366s) significantly faster than *standard* (mean 453s). There was also a significant interaction effect between Interface and Task Pairing ($F_{1,8} = 17.0$, $p = 0.03$). Figure 6.3 depicts the nature of this interaction effect, suggesting that while *commandmap* was faster overall, the benefit was more pronounced for Task One than Task Two. Participant comments suggested that Task Two may have been easier with the Ribbon interface than Task One, with one noting that Task Two required fewer tab switches (the sequence [*Word Count*, *Insert Comment*] requires a tab switch while [*Word*

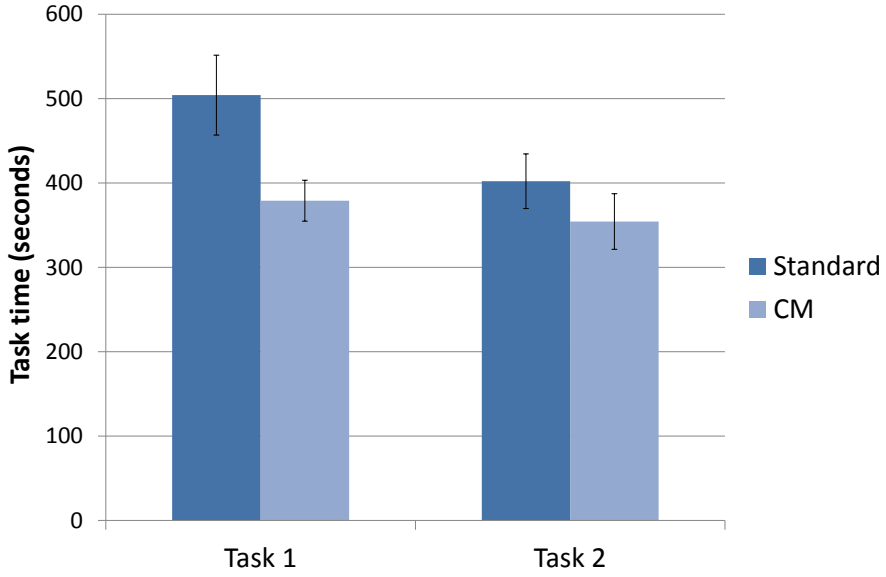


Figure 6.3: Mean task times in Experiment 6A. Error bars show standard error.

Count, *Footnote*] does not, leading to 78 tab switches in Task One compared to 50 in Task Two).

This explanation for the difference in completion times between tasks is supported by the results of Chapter 5, which showed that user performance with the Ribbon is highly dependent on the number of tab switches involved. Examining performance on each task individually using between-subjects analyses, there was a significant effect of Interface in Task One ($F_{1,8} = 5.5$, $p = 0.046$), but none in Task Two ($F_{1,8} = 1.1$, $p = 0.3$). This provides partial support for **H1**. An alternative explanation of the interaction is that asymmetric skill transfer occurred – although it might be argued that the CommandMap was particularly good preparation for the Normal user interface in Task Two (explaining the larger improvement across Tasks One and Two with Normal than with CommandMaps), the previous explanation seems more compelling. Further, even if the explanation of asymmetric skill transfer is correct, then this would suggest a further and unexpected benefit of CommandMaps in helping users learn standard interfaces.

	Standard	CommandMap	Wilcoxon z	Significance (p)
Mental demand	2.8 (0.9)	2.6 (1.0)	0.4	0.34
Physical demand	**3.2 (1.1)**	**1.9 (0.9)**	**2.2**	**0.01**
Temporal demand	**3.1 (1.1)**	**2.2 (0.6)**	**1.9**	**0.03**
Success	4.2 (0.4)	4.3 (0.5)	0.0	0.5
Hard work	**3.2 (1.2)**	**2.1 (0.9)**	**1.9**	**0.03**
Frustration	**2.5 (1.4)**	**1.6 (0.8)**	**1.8**	**0.04**

Table 6.2: Mean (standard deviation) NASA-TLX responses (1 = low, 5 = high). Rows in bold indicate significant differences.

Question (1 = Disagree, 5 = Agree)	Before CM [mean (s.d.)]	After CM [mean (s.d.)]	z	p
Q: It will be/was easy to find controls in the full-screen interface.	3.8 (0.6)	4.0 (0.7)	0.7	0.2
Q: I will be/was able to remember the locations of controls in the full-screen interface.	**3.5 (0.9)**	**4.4 (0.7)**	**2.3**	**0.01**
Q: The full-screen interface is/was visually overwhelming.	3.2 (0.9)	2.9 (1.0)	0.8	0.2
Q: I will be/was able to interact quickly with the full-screen interface.	**3.5 (1.0)**	**4.4 (0.8)**	**2.5**	**0.01**
Q: I would be discouraged from using software with this kind of interface in the future.	**2.0 (0.9)**	**1.6 (0.8)**	**1.6**	**0.05**

Table 6.3: Likert-scale questions and responses before and after CommandMap use in Experiment 6A. Significant differences in bold. z indicates Wilcoxon z score.

Subjective responses

NASA-TLX responses (Table 6.2) showed that participants found CommandMaps to be less physically demanding, less temporally demanding, easier to use and less frustrating than the standard interface. Subject preferences were also in favour of CommandMaps, with nine out of ten users preferring CommandMaps and one preferring the standard interface ($\chi^2 = 4.9$, $p = 0.03$); this provides support for **H2**.

Results for the CommandMap questionnaire indicated that participants' opinions of the CommandMap improved after they were given the opportunity to use it. In particular, participants underestimated their ability to remember control locations and interact quickly with the CommandMap (Table 6.3). This result was

reinforced by subjects' comments, for example: *"at first glance I thought that the interface was too busy to find items. But later I found that I could remember icons exactly without the need to look for them. I could use the interface when I pushed* `Ctrl` *so the interface was not on the screen all the time and it was easy to push or release the* `Ctrl` *button."* There was also a marginal effect ($p = 0.05$) in the change of response to the question *I would be discouraged from using software with this kind of interface in the future*. Initially, the responses expressed disagreement (mean 2.0, s.d. 0.9); and after using the system their level of disagreement was stronger (mean 1.6, s.d. 0.8). There was no change in perceived ease of finding controls, or the degree to which the interface was seen as visually overwhelming.

6.3 Experiment 6B: Realistic usage in Word and Pinta

Experiment 6A showed that user performance and subjective preferences for CommandMaps exceeds that of the Ribbon when executing repetitive tasks that are interwoven with realistic activities on a document. This extends prior results showing that command selection time (without any associated task) is reduced with CommandMaps. Experiment 6B moves further toward real-world use, adding the following elements of realism.

1. **Different application contexts.** CommandMaps are designed to be a general technique, so Experiment 6B examines both word-processing (Microsoft Word) and painting (Pinta) applications.

2. **Realistic learning curve.** Previous CommandMap studies have all used highly repetitive selections of a small number of commands within short duration experiments. Experiment 6B, in contrast, examines a week of daily skill development in a more realistic task. Further, to gain better insight into novice interaction (and unlike prior studies), no explicit initial training was given.

3. **Less repetitive tasks.** Unlike previous studies, most of the commands used in Experiment 6B were executed only once per task, and there was a wider

range of commands. Furthermore, the tasks were designed to intersperse typing and drawing activities between command selections.

4. **Open-ended tasks.** Experiment 6A gave participants a specific set of instructions, relieving the participant from the need to expend cognitive resources on issues such as task strategy; in Experiment 6B, in contrast, participants were given a target document or image and asked to reproduce it.

6.3.1 Participants and apparatus

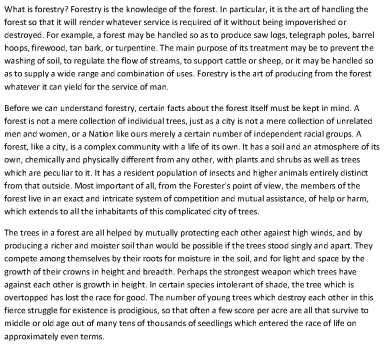
Twelve participants (10 male, 2 female; aged 21-34 years) were recruited from the University of Canterbury and the University of Saskatchewan. None had been involved in Experiment 6A, nor had any previously seen a CommandMap. Apparatus was the same as Experiment 6A.

6.3.2 Procedure

Participants were assigned four tasks to complete on each of five successive days. Two tasks were completed using Word, and two using Pinta. One of the tasks with each application was performed using the CommandMap version of the software, and the other task was completed using the application's standard interface. To facilitate rigour in the analysis of skill development over the week, the experimental task with each interface was held constant for each of the five days. All participants completed the two tasks with each application in the same order, but the order of the applications and of their versions (standard and CommandMap) was counterbalanced between participants. On each day, participants spent five to ten minutes completing each of the four tasks.

For each Word task, participants were provided with an on-screen document containing text, as well as a printout of the intended formatting (see Figure 6.4). In the Pinta tasks, participants were given an initial image and had to apply filters and use drawing tools to replicate a target state shown on a printout. Participants were instructed to complete tasks in any way they wished, and to click 'Start' and 'Stop' buttons at the beginning and end of the task.

Participants completed a demographics questionnaire before the experiment,

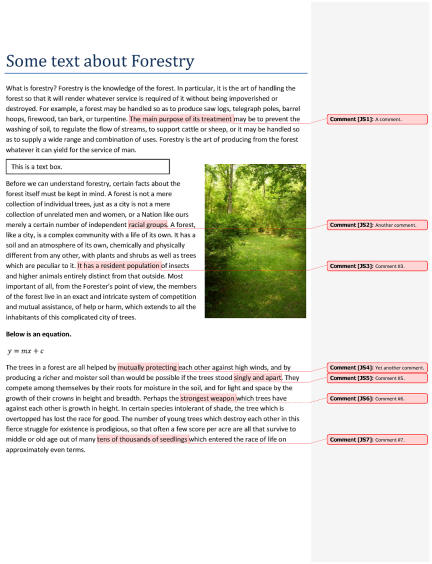


What is forestry? Forestry is the knowledge of the forest. In particular, it is the art of handling the forest so that it will render whatever service is required of it without being impoverished or destroyed. For example, a forest may be handled so as to produce saw logs, telegraph poles, barrel hoops, firewood, tan bark, or turpentine. The main purpose of its treatment may be to prevent the washing of soil, to regulate the flow of streams, to support cattle or sheep, or it may be handled so as to supply a wide range and combination of uses. Forestry is the art of producing from the forest whatever it can yield for the service of man.

Before we can understand forestry, certain facts about the forest itself must be kept in mind. A forest is not a mere collection of individual trees, just as a city is not a mere collection of unrelated men and women, or a Nation like ours merely a certain number of independent racial groups. A forest, like a city, is a complex community with a life of its own. It has a soil and an atmosphere of its own, chemically and physically different from any other, with plants and shrubs as well as trees which are peculiar to it. It has a resident population of insects and higher animals entirely distinct from that outside. Most important of all, from the Forester's point of view, the members of the forest live in an exact and intricate system of competition and mutual assistance, of help or harm, which extends to all the inhabitants of this complicated city of trees.

The trees in a forest are all helped by mutually protecting each other against high winds, and by producing a richer and moister soil than would be possible if the trees stood singly and apart. They compete among themselves by their roots for moisture in the soil, and for light and space by the growth of their crowns in height and breadth. Perhaps the strongest weapon which trees have against each other is growth in height. In certain species intolerant of shade, the tree which is overtopped has lost the race for good. The number of young trees which destroy each other in this fierce struggle for existence is prodigious, so that often a few score per acre are all that survive to middle or old age out of many tens of thousands of seedlings which entered the race of life on approximately even terms.

(a) Word Task One (before).



Some text about Forestry

What is forestry? Forestry is the knowledge of the forest. In particular, it is the art of handling the forest so that it will render whatever service is required of it without being impoverished or destroyed. For example, a forest may be handled so as to produce saw logs, telegraph poles, barrel hoops, firewood, tan bark, or turpentine. The main purpose of its treatment may be to prevent the washing of soil, to regulate the flow of streams, to support cattle or sheep, or it may be handled so as to supply a wide range and combination of uses. Forestry is the art of producing from the forest whatever it can yield for the service of man.

Comment [351]: A comment.

This is a text box.

Before we can understand forestry, certain facts about the forest itself must be kept in mind. A forest is not a mere collection of individual trees, just as a city is not a mere collection of unrelated men and women, or a Nation like ours merely a certain number of independent racial groups. A forest, like a city, is a complex community with a life of its own. It has a soil and an atmosphere of its own, chemically and physically different from any other, with plants and shrubs as well as trees which are peculiar to it. It has a resident population of insects and higher animals entirely distinct from that outside. Most important of all, from the Forester's point of view, the members of the forest live in an exact and intricate system of competition and mutual assistance, of help or harm, which extends to all the inhabitants of this complicated city of trees.

Comment [352]: Another comment.

Comment [353]: Comment #3.

Below is an equation.

$$y = mx + c$$

The trees in a forest are all helped by mutually protecting each other against high winds, and by producing a richer and moister soil than would be possible if the trees stood singly and apart. They compete among themselves by their roots for moisture in the soil, and for light and space by the growth of their crowns in height and breadth. Perhaps the strongest weapon which trees have against each other is growth in height. In certain species intolerant of shade, the tree which is overtopped has lost the race for good. The number of young trees which destroy each other in this fierce struggle for existence is prodigious, so that often a few score per acre are all that survive to middle or old age out of many tens of thousands of seedlings which entered the race of life on approximately even terms.

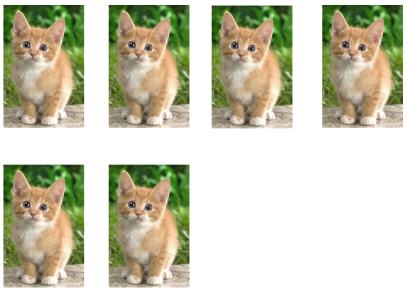
Comment [354]: Yet another comment.

Comment [355]: Comment #5.

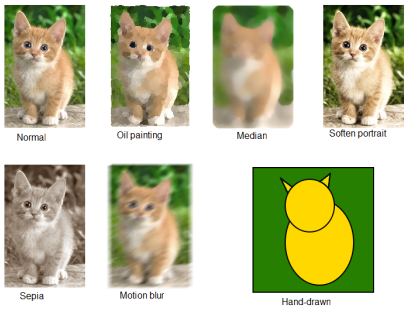
Comment [356]: Comment #6.

Comment [357]: Comment #7.

(b) Word Task One (after).



(c) Pinta Task One (before).



(d) Pinta Task One (after).

Figure 6.4: Initial and target states for two of the tasks in Experiment 6B.

and filled out NASA-TLX [80] worksheets after each task. Additional questionnaires probing participants' opinions of CommandMaps were given at the beginning and end of the first day (Session One) and at the end of Session Five. The participants were interviewed after completing all tasks on the fifth day. All sessions were videotaped.

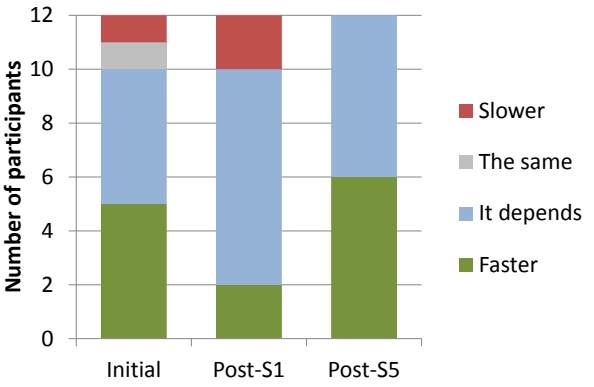
6.3.3 Results

Experiment 6B was intended to answer research questions **Q2** through **Q5**. Below, the results of Experiment 6B are presented in terms of those questions, drawing on quantitative performance and preference measures as well as qualitative data from the video transcripts, questionnaires and post-study interviews.

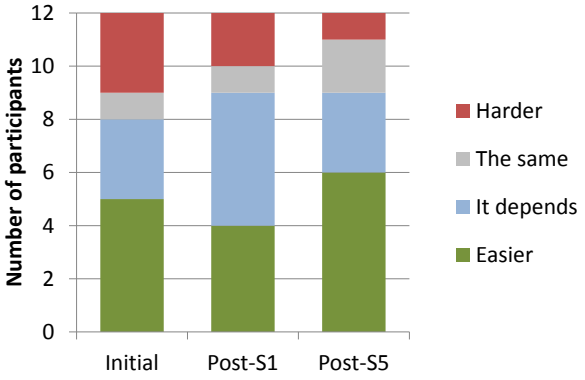
Q2. How do users respond when first using CommandMaps?

Initial reactions to the Word and Pinta CommandMaps were mixed (see Figures 6.5 and 6.6, leftmost bars). One participant expected the CommandMap to slow his performance, but almost all others thought they would be faster at least some of the time. Measures of anticipated ease of use and confusion were less favorable, with three users anticipating the CommandMap to be harder than the normal interface, and five users rating it as initially more confusing. Eight participants commented that when they first saw the CommandMap, they found it overwhelming; but all of those participants said that their initial impression changed after only one or two five-minute sessions. For example, P3 said *"when I first pulled it up, it was like 'whoa'... but then you quickly figure out the categories and then it's pretty quick."* As seen in Figure 6.6, most participants initially preferred the normal interface for Word, though six participants explicitly mentioned that the main factor in their decision was their existing familiarity with the Word interface. In comparison, with Pinta – an application that participants were less familiar with – the majority of participants (7/12) initially preferred the CommandMap.

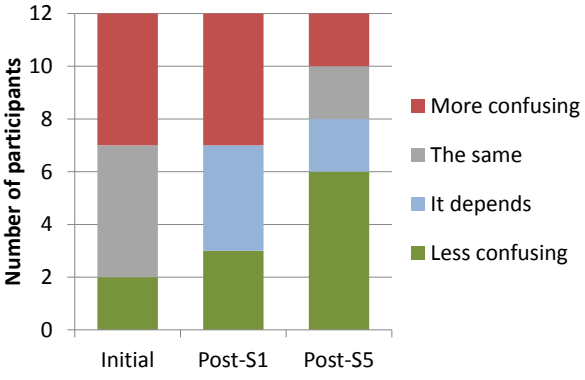
Despite initial impressions, six participants commented in interviews that CommandMaps would be most advantageous for new users, since they provide an overview of an application's functionality. P1 thought that the main advantage of CommandMaps was *"getting to know the commands that are in a new application quickly"*, and P4 stated that *"one of the advantages is showing options that you*



(a) Faster/slower



(b) Easier/harder



(c) More/less confusing

Figure 6.5: Participants' opinions before the experiment, after Session One, and after Session Five, on whether CommandMaps are faster or slower, easier or harder to use, and more or less confusing than the standard interfaces.

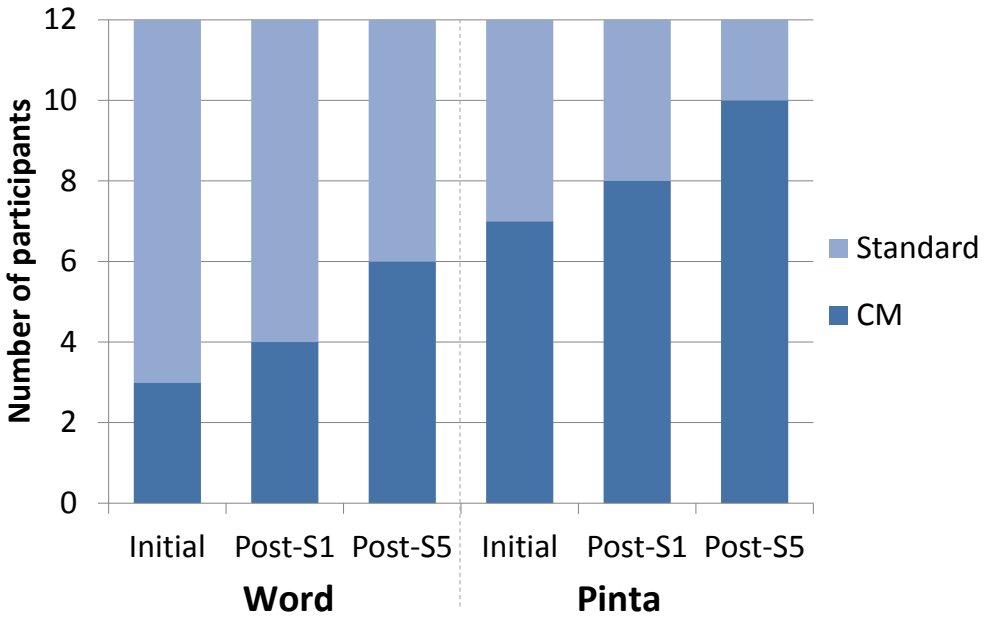


Figure 6.6: Participants' interface preferences for Word and Pinta throughout Experiment 6B.

maybe might not have known about". P1 indicated that he would recommend the interface to his mother, a novice Word user: *"I'd say 'just hold the Ctrl key and look, and you'll find it in there.' And that would help her, I think."*

Two participants also commented that they were able to transfer their existing spatial knowledge of the Ribbon to the Word CommandMap: P8 stated that *"I could apply my existing knowledge to the layout of the CommandMap, so I knew what section it was under and roughly how far along it was."*

Q3. How do performance and subjective impressions change in real tasks as users gain experience?

As expected, performance improved throughout the week in all four interfaces (Figure 6.7). Separate 2×5 ANOVAs were run on the Word and Pinta data for within-subjects factors Interface (*standard*, *commandmap*) and Session (*1 – 5*). As Figure 6.7 suggests, neither analysis showed an effect of Interface (Word: $F_{1,11} = 1.37$, $p = 0.27$; Pinta: $F_{1,11} = 0.01$, $p = 0.93$), which is unsurprising given that participants spent proportionally more time manipulating content and less

time selecting commands compared to Experiment 6A. As expected, both analyses showed a significant effect of Session (Word: $F_{4,44} = 29.28$, $p < 0.001$; Pinta: $F_{4,44} = 69.98$, $p < 0.001$) due to the increase in performance over time, and neither analysis showed an Interface×Session interaction (Word: $F_{4,44} = 0.39$, $p = 0.81$; Pinta: $F_{4,44} = 0.07$, $p = 0.99$).

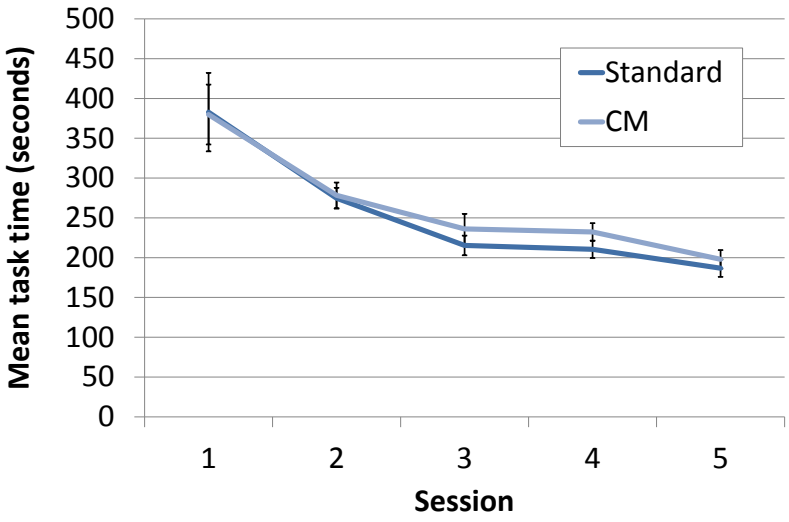
Subjective impressions of the CommandMaps improved through the week (Figures 6.5 and 6.6). By the end of the week, 10/12 participants preferred CommandMaps for Pinta ($\chi^2 = 4.1$, $p = 0.04$), but preferences were evenly split for Word. Interview comments showed that several participants initially thought the CommandMap would be difficult to use, but changed their minds after trying it. Five participants mentioned that they liked the Pinta CommandMap because it removed submenus: *"once you get to know it, it's really fast. Like if you know the blur button is going to be here [points], it's basically Ctrl and then move your mouse and it's almost instantaneous, instead of going through the submenus."* Two participants also liked the full-screen workspace with CommandMaps.

Q4. Do CommandMaps generalise to multiple application contexts?

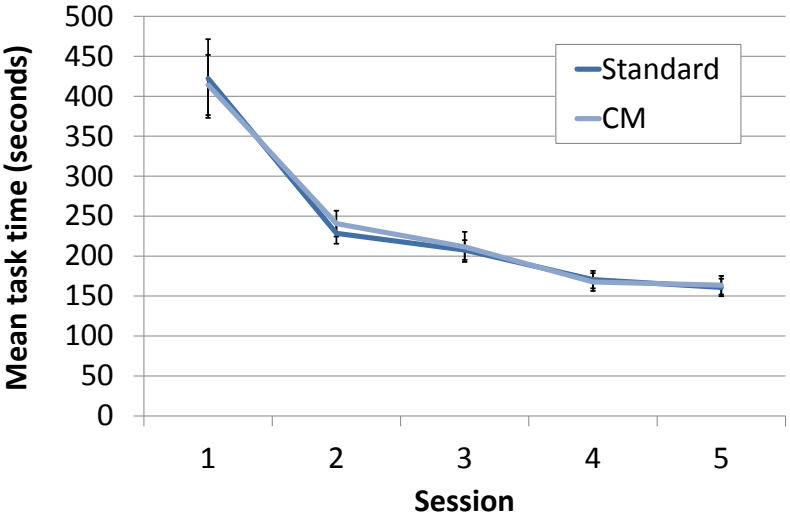
Figure 6.6 shows that CommandMap preferences were stronger in Pinta than Word, with familiarity with Word's Ribbon being mentioned by six participants in explaining their preferences. The two participants who maintained a preference for Pinta stated that its menu hierarchy made it easier to find commands compared to the CommandMap's parallelised layout. The perceived advantages of CommandMaps were the speed of accessing commands (five participants), not having to scroll through Ribbon tabs (one participant), and the ability to see an overview of all commands in the application (six participants).

Q5. What usability issues arise with CommandMaps in real-world use?

Seven participants mentioned that they had problems using the `Ctrl` key to activate the CommandMap. In particular, the CommandMap would briefly appear when issuing hotkey shortcuts: *"Once I finish a sentence, I automatically save. It's nothing I even think about, and [the CommandMap] flickers on screen."* Three participants also noted that they would prefer the activation key to act as a toggle, rather than needing to hold it for continual display.



(a) Word



(b) Pinta

Figure 6.7: Mean task times for each session in Experiment 6B.

Nine participants mentioned that item layout was critical to both interfaces' success, with three stating that the Word Ribbon categories were poorly-named. Four participants indicated that the Pinta CommandMap needed an improved layout, and P3 suggested placing category labels on the left-hand side of the CommandMap to improve category identification. Several participants disliked having Pinta's toolbox in the CommandMap: P5 said that *"when you're rapidly changing between different tools, I don't think CommandMaps are useful... I think you should be able to determine based on frequency which commands should be in a CommandMap and which should be permanently available"*.

Prior to the study, there was a concern that the modal appearance of the CommandMap, hiding nearly the entire view of the work surface, might disorient users and interfere with their task flow. However, apart from flicker when using keyboard shortcuts, only two participants mentioned this as an issue. Five participants explicitly said it was not a problem, with several mentioning that it was better than the standard interface; P8 said *"I think it was more effective,"* P4 said *"with Word, you have to go to a different tab, and go back and do something, which can be a bit disorienting... whereas with this it's always the same thing,"* and P12 said *"You were kind of still looking in the same place as well, because it's right in the middle of the screen. I found I didn't lose track of where I was, whereas if I went back to the menu and had to go back down again I'd find that more distracting"*. The semi-opaque CommandMap display also received positive comments: P3 said *"I liked with the Pinta one, it was kind of transparent, so you could push Ctrl and still see what's underneath."*

6.4 Discussion

To recap, Experiment 6A used a CommandMap version of Microsoft Word to demonstrate that CommandMaps have performance advantages over standard interfaces in a task that involved interweaving command invocation with actions on the work surface. Experiment 6B analyzed the evolution of CommandMap performance and perceptions over the course of a week, revealing a number of advantages and disadvantages. Overall, the results are extremely positive for CommandMaps: most participants immediately preferred the Pinta CommandMap to traditional interface controls, and 10 of 12 preferred it after five days of use; and

half of the participants preferred the Microsoft Word CommandMap to the highly refined and familiar Ribbon interface after just a few sessions of use.

The sections below discuss potential concerns regarding the limitations of the methodology, as well as the issues involved with creating future real-world implementations of CommandMaps.

6.4.1 Methodological limitations

Although the experiments presented in this chapter were designed to simulate real-world use as closely as possible, trade-offs were still made in order to measure certain properties of performance. For example, users in the real world would be unlikely to perform the same editing tasks in the same documents every morning for five days, but this study design was necessary in order to gauge user learning in a reliable manner and without using up too much participant time.

Ideally, in order to truly measure CommandMap usage in the wild, a field study would be performed. Conducting such a study involves a number of challenges. First, fully functional systems must be engineered and deployed; Section 6.4.2 discusses this in greater depth. Second, accurate measurement techniques must be developed. Measurements of task time in the wild can be difficult due to segmentation issues (i.e., how does the system know when a command selection, or a task, starts and ends?) Parts of this problem are solvable: the Input Observer [52] is able to measure pointing times by detecting ballistic mouse movements, but determining when the user starts searching for a command is a much more difficult task. Overcoming these challenges is necessary before a field study can be completed, and these are therefore left for future work.

6.4.2 Technical limitations

As discussed above, a field study on CommandMaps would be an ideal test of their real-world suitability. However, in order to properly study the use of CommandMaps 'in the wild', the technique would ideally be integrated with widely used software. Our experiences with developing a CommandMap version of Word suggest that achieving this integration can be difficult due to the lack of extensibility and support for modification in many commercial applications. For example, the Word Object Model API provided access to limited functionality, prohibiting

a complete replication of the original Ribbon interface. Pinta, in contrast, was much easier to extend, but its small user base would limit the value of its use in a field study. Future CommandMap implementations are therefore probably most feasible in open-source applications which have large user bases.

In general, however, in order to provide support in future for CommandMaps and similar HCI research, it would be useful if UI toolkits allowed manipulation of control and panel locations, as well as providing better support for programmatically accessing interface functionality. Metisse [25] is one example of a partial workaround for the research community, although it only works on the X window system. Prefab [43] is another example of a research solution for reverse engineering interface controls, though it does not allow programmatic access to functionality.

6.5 Conclusions

The CommandMap is a user interface command selection technique that makes all, or nearly all, commands available to the user in a single display that is overlaid on the workspace when a modifier key is pressed. A previous study (Chapter 5) demonstrated the potential of CommandMaps by showing that the time to acquire a CommandMap item in response to a cued stimulus is faster than with menus or ribbons. However, critical questions remained regarding their practicality, particularly because of the workflow disruption that might occur due to their displacement of the view of the workspace. To examine this and other issues regarding the practicality of CommandMaps, this chapter presented two real-world implementations of CommandMaps for Microsoft Word and Pinta. Experiment 6A showed advantages of CommandMaps over the Microsoft Word Ribbon in tasks that involved interleaved access to the workspace and the CommandMap. Experiment 6B showed how users' perceptions of CommandMaps rapidly evolve over the first week of use, as well as highlighting their advantages and opportunities for improvement.

In particular, many participants reported visual overload when first using the CommandMap, suggesting that there are unexplored opportunities to improve novice use. Chapter 7 therefore describes the StencilMap, an augmentation of the CommandMap intended to reduce visual complexity and improve novice per-

formance.

Chapter 7

StencilMaps: A Spatially Consistent Subset Interface for Training Novice Users

Chapters 5 and 6 presented and evaluated the CommandMap interface, designed to improve expert point-and-click command selection performance. While experiments demonstrated advantages for expert users, results for novice users were mixed – for example, Experiment 5B showed that the standard Ribbon interface performed just as well as a CommandMap for novices (see Figure 5.11), and the majority of participants in Experiment 6B mentioned an initial impression of visual overload (see Section 6.3.3).

Although the large number of items displayed simultaneously in a CommandMap reduces the mechanical actions required to select an item, if a user does not already have knowledge of that item's location, it requires them to search through a larger item set in order to find the one they want. In order to reduce the time required to visually search for items, this chapter investigates the use of *subset interfaces*, which assist novice interaction by emphasising a small subset of relevant items.

Many subset interfaces already exist, and they can be divided into two categories: *dynamic* and *static*. Dynamic subsets change automatically and adapt to user behaviour (e.g., ephemeral adaptation [57]). Static subset interfaces can be further subdivided into those that are pre-defined by the system designer (e.g., Training Wheels [23]) and those with subsets chosen by the user (e.g., AdaptableGIMP [118] or the Quick Access toolbar in Microsoft Word). Such interfaces are often used to assist novice interaction. For example, to assist novice users in learning the rich functionality of a graphics application, the AdaptableGIMP [118] allows community members to create static command subsets for common tasks, such as red-eye reduction or adding a sepia effect to an image. When se-

lected by the user, the task-based subsets replace the toolbox used in the standard interface, bringing all of the commands needed for a task within easy reach.

This chapter focuses on the use of *static* subsets to assist novice interaction. Previous research with such interfaces (e.g., [23, 55, 142]) shows that working with subsets can improve both performance and satisfaction for novice users. However, less is known about the best way of presenting command subsets to the user. In addition to the palette-based approach of AdaptableGIMP, researchers have created systems that remove unneeded commands and condense the remaining interface elements [142], or that selectively highlight or disable items [23, 67].

Different approaches to supporting subset interfaces vary in the degree of separation between the presentation of the subset interface and the presentation of the standard interface. For example, when a user selects a task subset in AdaptableGIMP, the subset is placed within a new palette, and the icons in the palette have no spatial relationship to their location in the standard UI. This display separation between the subset and the standard interface creates a potential problem for the user, because selections from within the subset do not assist learning of the standard user interface. The user's efficiency with the subset, therefore, is dependent on the desired target item being present in the subset; and when it is absent users must search the standard user interface that is less familiar than it would have been if all previous selections had been made within it.

Some subset presentation techniques (e.g., [57, 59, 67]) avoid the problem of separately displaying the subset and the standard interface by keeping subset items in the same location as the full interface, and visually emphasising the subset items. However, evaluations of this kind of interface have had mixed results (see Section 4.3.3). Specifically, highlighting interfaces such as these present two risks. First, users may not attend to the highlighting if it is not salient enough – in this area, stencils [59] have shown more promising results than colour-based highlighting [67]. Second, if the highlighting significantly decreases the difficulty of search, the reduced effort required on the part of the user may mean that they do not learn item locations as well as they otherwise would have (see Section 3.3 for a discussion of the role of effort in spatial learning). The experiments performed in this chapter are designed to further examine these issues.

This chapter presents and studies the idea of emphasizing a subset in a CommandMap by adding a dark semitransparent 'stencil' overlay (Figure 7.1). Sten-

cils have been used previously for providing interface tutorials [111] and search in file browsers [59], and a related approach has been used to visualise application usage [138], but they have not yet used to create command subsets for novice users.

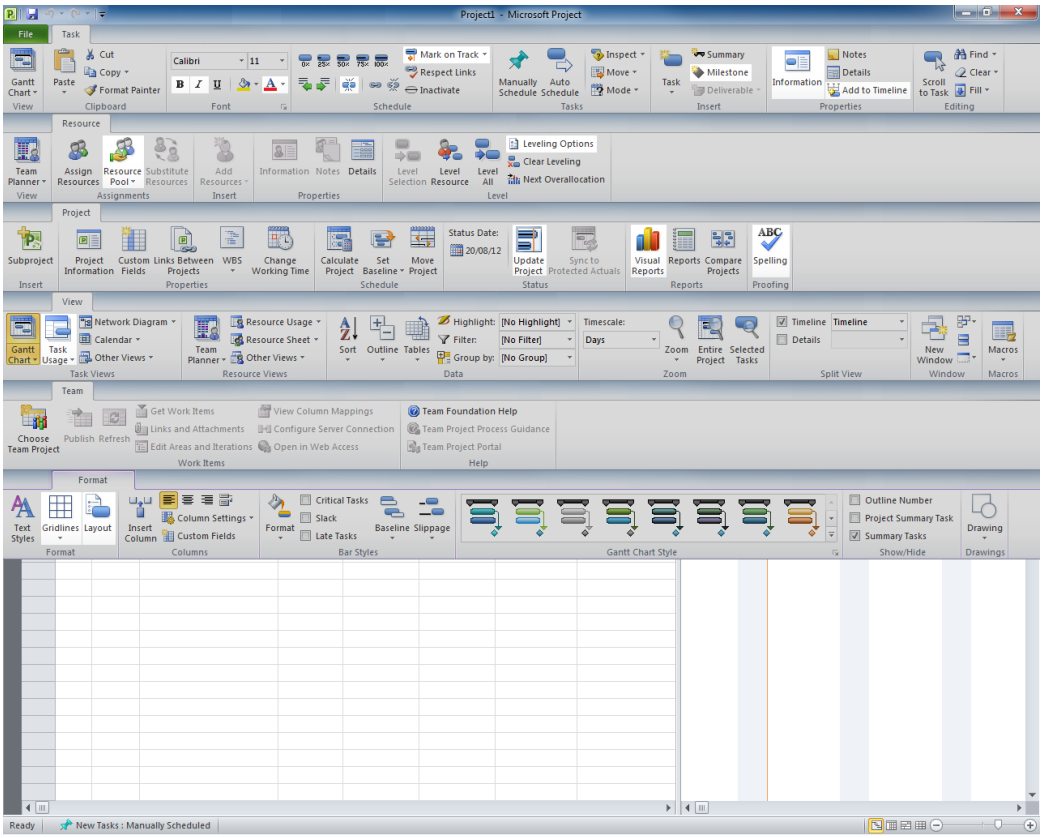


Figure 7.1: A StencilMap in Microsoft Project 2010.

This chapter presents the design of the new interface, called StencilMaps, and the results of two laboratory experiments investigating StencilMaps' performance both in terms of decreasing visual search time and allowing transfer of knowledge from the subset (i.e., 'stencilled') interface to the full interface.

7.1 Design of StencilMaps

The principles of subsets, stencils, and spatial memory guided the StencilMap design. The StencilMap (Figure 7.1) is built on top of the CommandMap interface; although this type of UI is not a requirement for the stencil approach, the

command visibility of CommandMaps makes it easier for novices to see all of the items in the subset, and the lack of hierarchy allows for a high performance ceiling once users have learned item locations.

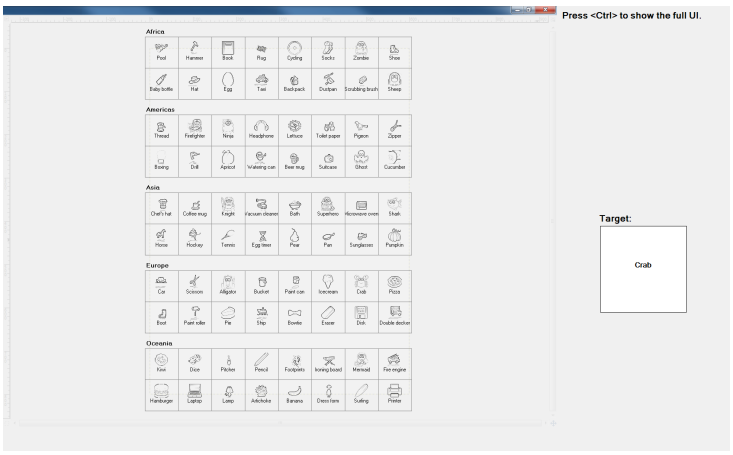
A StencilMap consists of a CommandMap with an overlaid translucent mask, which emphasizes salient controls and reduces the visual impact of unimportant commands. Like a normal CommandMap, the StencilMap is hidden by default, and is activated by holding down a hotkey (such as `Ctrl`) or by clicking an on-screen button.

The performance characteristics of StencilMaps are based on two ideas. First, the stencil is intended to reduce the visual search time needed for novices to find a command (i.e., visual search is proportional to the number of items in the search, and stencil-based emphasis should greatly reduce this number). This is particularly important for CommandMaps, since the large number of visible commands can present a barrier for novice use. Second, StencilMaps do not change the location of any commands from their positions in the full interface (i.e., commands are located in the same position as in the normal CommandMap without the stencil). This means that any spatial location memory built up through use of a subset can be transferred to the full UI.

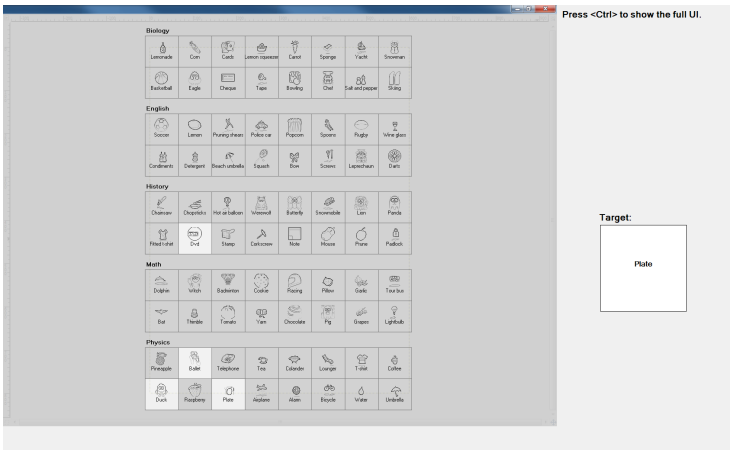
StencilMaps, therefore, seek to support novice users by visually promoting only a relevant subset of the application's functionality. At the same time, StencilMaps provide users with the opportunity to gain interface-level expertise through in-situ learning of command locations. At a higher level, developing application expertise also involves gaining knowledge of what the commands do and their role in completing different tasks, but this type of high-level expertise development is beyond the scope of this work (see Appendix C for a review).

7.2 Experiment 7A: CommandMaps, StencilMaps, and ephemeral highlighting

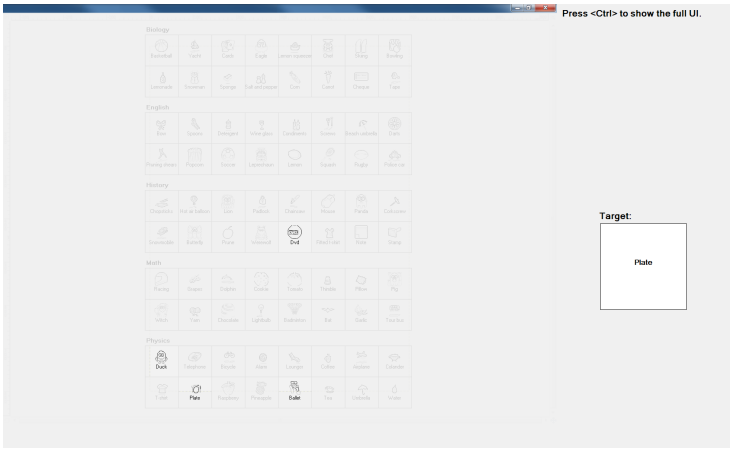
Experiment 7A was designed to answer two questions: first, to determine whether highlighting could improve performance in a CommandMap; and second, to compare the performance of StencilMaps against ephemeral adaptation [57], a technique that shows predicted items immediately (abrupt onset) and fades in non-predicted items after a short delay (gradual onset). Since this experiment deals



(a) CommandMap



(b) StencilMap



(c) Ephemeral Highlighting

Figure 7.2: The three interface conditions in Experiment 7A.

with static, rather than adaptive, subsets, our implementation of the ephemeral adaptation technique is henceforth referred to as 'ephemeral highlighting.' (Note that 'highlighting' here refers to relative highlighting, by darkening or fading out the out-of-subset interface items.)

7.2.1 Experimental conditions

The experiment compared performance with a StencilMap (Figure 7.2b) to that with a CommandMap (Figure 7.2a), and to a CommandMap with ephemeral highlighting (Figure 7.2c). The CommandMap was included in order to isolate any performance benefit stemming from the layout of the CommandMap from that arising from stencil-based highlighting. The ephemeral highlighting condition (Figure 7.2c) was included as a representation of the state of the art in subset highlighting.

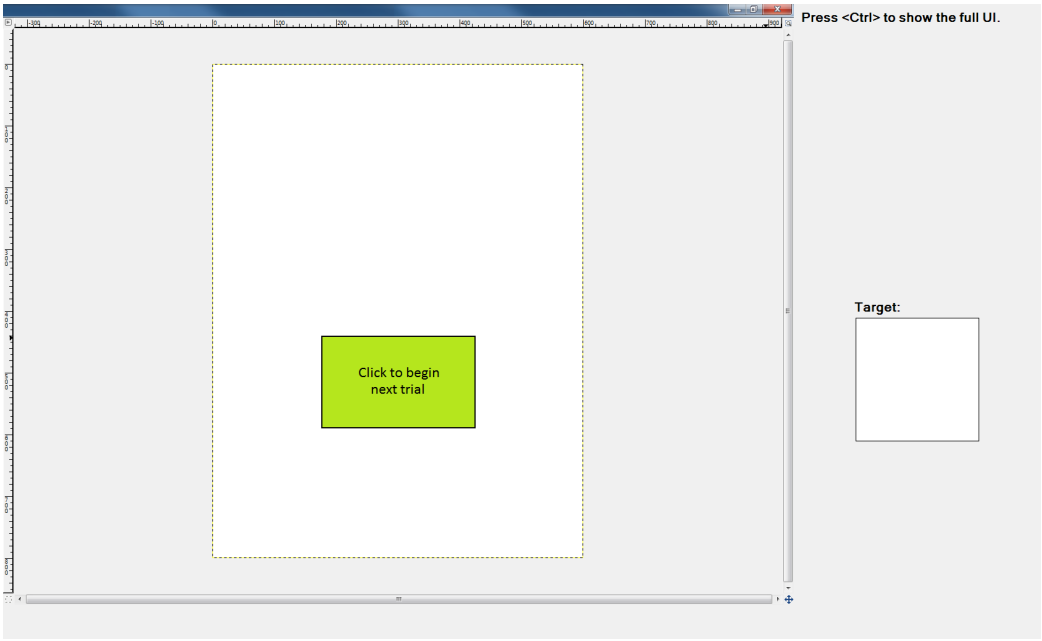


Figure 7.3: The study interface used in Experiment 7A.

CommandMaps. The baseline technique showed all 80 commands of the interface at once when the user pressed `Ctrl` (Figure 7.2a). The commands were shown until the user selected a command with the mouse.

Ephemeral highlighting. This method was based on Findlater et al.'s *ephemeral adaptation* [57] in which predicted menu items are shown immediately (to promote their visual prominence and capture the users' attention [104, 211]), while unpredicted ones are faded in after a short delay. When the CommandMap was invoked with the `Ctrl` key, the ephemeral technique showed the in-subset items instantly, while the out-of-subset items gradually faded in to view (Figure 7.2c). The implementation used in this experiment was based on the original design [57], but adapted to work with icon sets rather than menus, and to work with larger subsets. The 500ms onset time used in the original technique was shown in pilot studies to be too short for larger subset sizes. Since the time required to visually search a multi-item subset likely follows a linear trend [93], this implementation uses linearly increasing onset times: 500ms for subsets of size 4, 1000ms for subsets of size 8, and 2000ms for subsets of size 16.

StencilMaps. The StencilMap technique covers the out-of-subset items with a dark translucent overlay (Figure 7.2b), while in-subset items are shown normally (i.e., through 'holes' in the stencil). StencilMaps were implemented as described in the design section above.

7.2.2 Procedure

Participants completed a demographics questionnaire, and then performed a sequence of selection tasks in an experimental system similar to the one used in Chapter 5 (Figure 7.3). For each trial, a button was displayed with the text "Click to begin next trial". After the button was clicked, a textual stimulus was displayed on the right-hand side of the screen; the user had to hold down `Ctrl` in order to display the CommandMap, then select the target item. Each trial was timed from the appearance of the stimulus until successful target selection. Incorrect selections produced an audible beep. Participants were instructed to complete tasks "as quickly and accurately as possible".

To reduce unintended learning effects between conditions, three separate icon sets were used. Each icon set contained 80 items, and was randomly partitioned into five sub-categories. To avoid potential confounds caused by visual pop-out effects, the icons were all grayscale with a uniform visual style. All participants saw the icon sets in the same order, but order of interface was counterbalanced to

eliminate confounds caused by specific icon sets. Subsets and selection sequences for each icon set were determined using a one-off random process, and were the same for each participant. No targets were reused between sets.

Participants completed three consecutive phases with each interface: *familiarization*, *selection*, and *recall*. The *familiarization* phase consisted of 10 trials (data discarded), using different subsets and target items than in the main experiment. This was followed by two *selection* blocks of 25 trials, for each of the three subset sizes (4, 8 and 16). Each subset block contained five selections of each of five target items. Four out of five targets in each block were in the highlighted subset. This 4/5 (80%) subset accuracy level was chosen based on Findlater et al.'s results [57], which showed that at low accuracy (50%), ephemeral adaptation did not offer performance benefits over a full interface, likely because participants stopped attending to the highlights. Therefore, the study focuses on situations where highlighting has been shown to be useful (80% – consistent with Findlater et al.'s "High accuracy" condition).

Each interface finished with a *recall* test, where participants performed a single selection of each of the 30 targets used in that interface from a full, un-augmented CommandMap. This extra block was included to evaluate the longer-term effects of highlighting on spatial memory. Short breaks were given between blocks. Participants filled out a NASA-TLX questionnaire after each interface condition, and ranked the three interfaces at the end of the experiment.

7.2.3 Participants and apparatus

18 participants were recruited from a local university (8 male, 10 female; mean age 26.4 years) for the one hour experiment. It was performed on a Windows 7 PC with a 24 inch screen running at 1920×1200 resolution. The study software was written in C#, and recorded all experimental data, including selections and errors.

7.2.4 Design and hypotheses

The primary analysis used was a 3×3 RM-ANOVA on the subset blocks, with two within-subjects factors: Interface (*commandmap*, *ephemeral*, *stencilmap*), and Subset Size (*4*, *8*, *16*). Both factors were counterbalanced between partic-

ipants using a Latin square. A follow-up analysis was also planned, separating trials into in-subset and out-of-subset groups and performing separate 3×3 RM-ANOVAs on each group. Finally, a one-way ANOVA was planned on the recall blocks, with Interface as the only factor.

The primary hypotheses were as follows:

H1: StencilMap will be faster than both Ephemeral and CommandMap for in-subset selections. Since both StencilMaps and ephemeral highlighting should reduce the visual search space, both interfaces should provide quicker in-subset selections than the CommandMap. Additionally, the StencilMap should outperform ephemeral highlighting due to its permanent nature, and the fact that ephemeral highlighting allows the user only a limited amount of time to identify the target within the subset.

H2: Overall mean selection times for StencilMap will be faster than Ephemeral and CommandMap. As above, in-subset selections should be fastest for the StencilMap. In addition, StencilMaps is likely to perform the best overall – that is, the advantages of the StencilMap's quick in-subset selections should outweigh any potential slowdowns from out-of-subset selections.

H3: Users will subjectively prefer StencilMap over Ephemeral and CommandMap. Ephemeral highlighting may frustrate users if they cannot find items before the end of the onset timeout, and users may find it difficult to locate items in a CommandMap with no visual pop-out. It is expected that StencilMaps will therefore be subjectively rated higher than CommandMaps and ephemeral highlighting.

In addition, the study has the following (secondary) expectations:

E1: Overall mean selection times for Ephemeral will be faster than CommandMap. While ephemeral highlighting is not expected to provide as much of a performance benefit as the StencilMap, it should still be faster than the un-augmented CommandMap.

E2: CommandMap will perform better than both Ephemeral and StencilMap for out-of-subset items. Both ephemeral highlighting and StencilMaps encourage the user to visually search a subset of items first. If the item is not in that subset, the user may lose time on average compared to a CommandMap, where the user immediately begins searching the whole space.

E3: Performance on the recall test will be no different between the three conditions. Users should be able to learn item locations equally well in all three interfaces.

7.2.5 Results

For significant ANOVA results, partial eta-squared (η^2) is reported as a measure of effect size.

Overall error rates were low in all conditions (2.4% for *stencilmap*, 2.7% for *commandmap*, and 3.3% for *ephemeral*). The analysis of selection time below excludes trials with errors.

Selection times

As shown in Figure 7.4, mean selection times for the subset blocks were lower with *stencilmap* (2.52s, s.d. 0.79) than with *ephemeral* (3.51s, s.d. 1.42) and *commandmap* (4.36s, s.d. 1.20), giving a significant main effect of Interface ($F_{2,34} = 45.7$, $p < 0.001$, $\eta^2 = 0.73$). Pairwise Bonferroni-corrected comparisons (α = .05) showed that *stencilmap* was significantly faster than both *commandmap* and *ephemeral* overall. This result supports **H2**. *Ephemeral* was also significantly faster than *commandmap* (supporting **E1**). There was also a main effect of Subset Size ($F_{2,34} = 10.9$, $p < 0.001$, $\eta^2 = 0.39$), with size *4* (3.11s, s.d. 1.29) faster than *8* (3.67s, s.d. 1.44) and *16* (3.62s, s.d. 1.36). There was no significant interaction effect (Figure 7.5).

In-subset and out-of-subset selections

Separate analyses on selection time were performed for in-subset and out-of-subset targets (Figure 7.4). There was a significant main effect of Interface for in-

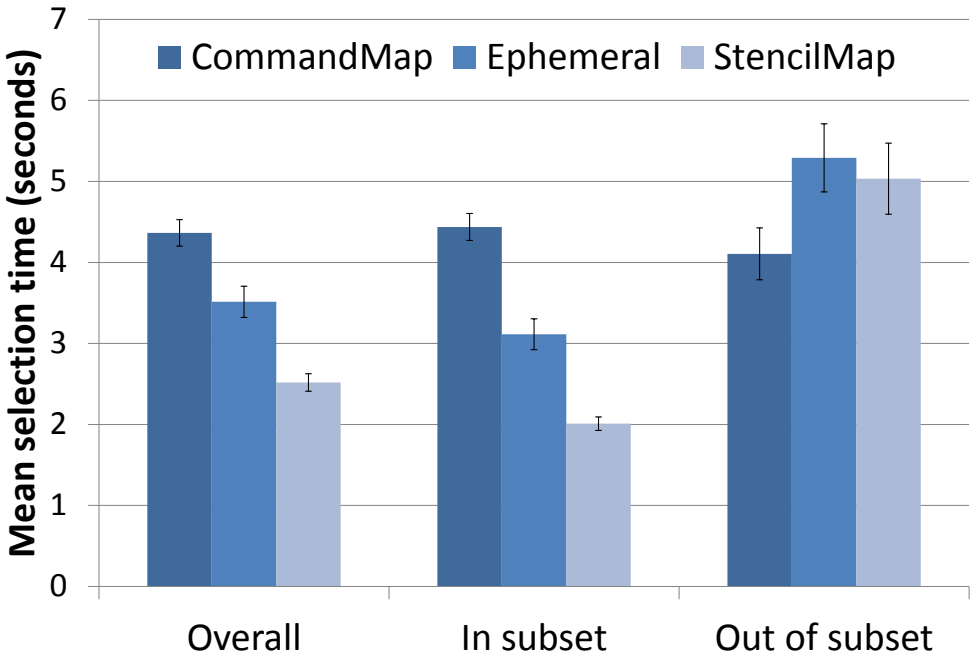


Figure 7.4: Overall mean selection times for Experiment 7A, and for in-subset and out-of-subset trials.

subset selections ($F_{2,34} = 63.6$, $p < 0.001$, $\eta^2 = 0.79$), with Bonferroni-corrected comparisons showing that *stencilmap* was significantly faster than both *epheme-ral* and *commandmap*. This supports **H1**. For out-of-subset selections, there was a trend suggesting a marginal effect of Interface ($F_{2,34} = 2.8$, $p = 0.07$), with an advantage for *commandmap*, supporting **E2**.

Learning

As shown in Figure 7.6, the greatest benefits of both types of highlighting are on the first selection of each highlighted item, when the user is completely unfamiliar with the item's location. *Stencilmap* was faster than *ephemeral* and *commandmap* throughout the repeated trials, although selections times approached convergence on the final trial.

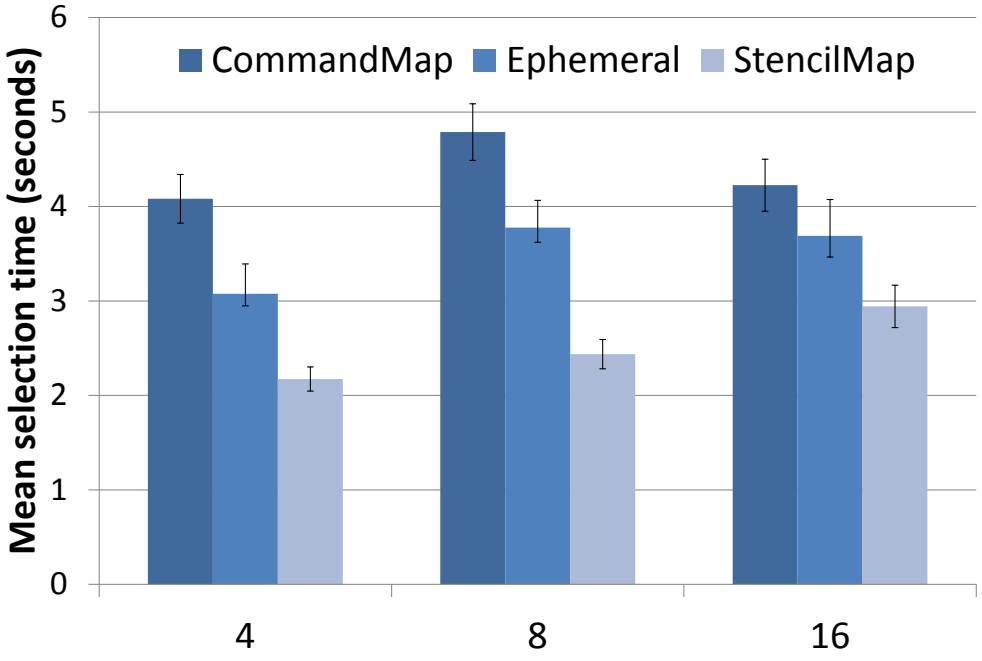


Figure 7.5: Mean selection times for each interface, broken down by subset size.

Recall times

As described above, the *recall* phase was performed on an un-augmented CommandMap after the main experiment, and measured participants' recall of the locations of items used in the experiment. A one-way ANOVA on overall recall times (Figure 7.7) showed a significant main effect of Interface ($F_{2,34} = 4.6$, $p = 0.02$, $\eta^2 = 0.21$), with *stencilmap* (4.44s, s.d. 2.00) slower than *ephemeral* (3.60s, s.d. 1.82) and *commandmap* (3.59s, s.d. 2.14). Pairwise Bonferroni-corrected comparisons showed *ephemeral* was no different from *commandmap* ($p = 0.96$), but a trend suggests that participants were faster at recalling items with *commandmap* than with *stencilmap* ($p = 0.053$). There was also a significant difference between *ephemeral* and *stencilmap*, with *ephemeral* being the faster of the two techniques ($p = 0.01$). Together, these results fail to support **E3**. This interesting result is explained in the Discussion (see Section 7.4.1).

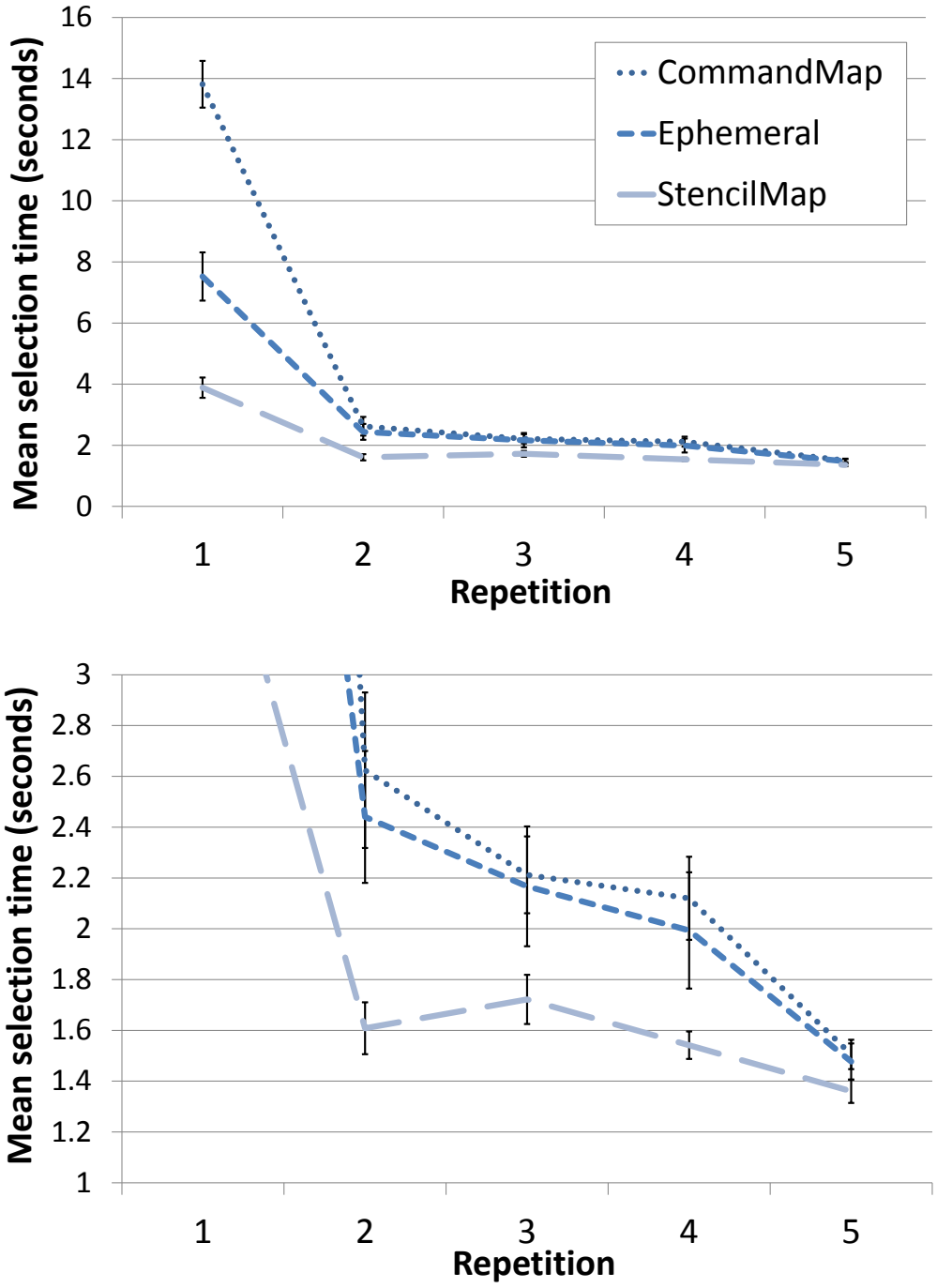


Figure 7.6: Mean selection time for in-subset targets, across repeated trials. The lower chart shows detail of repetitions 2-5.

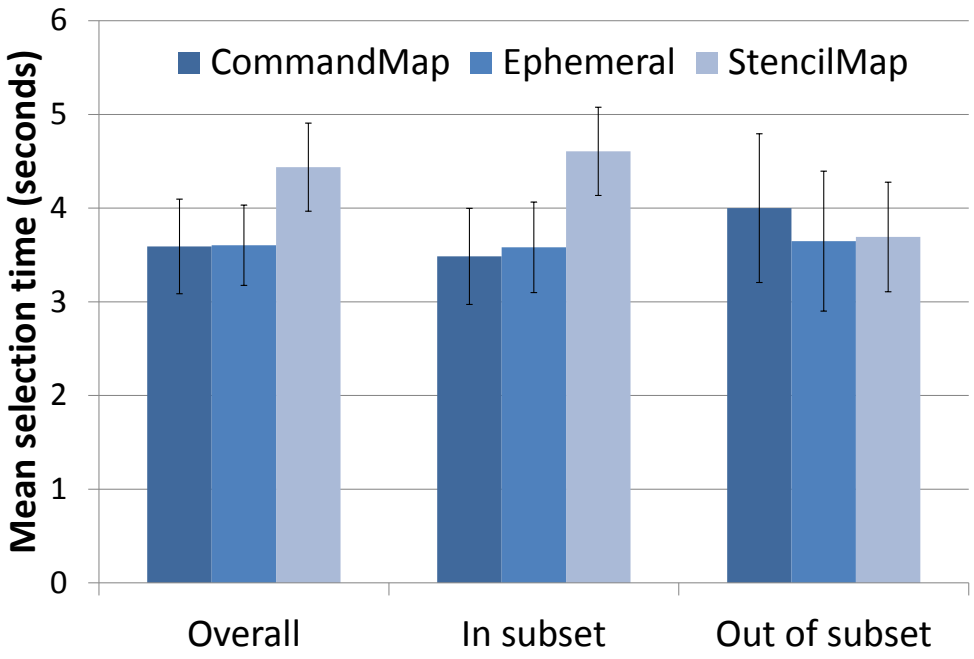


Figure 7.7: Overall selection times for the recall task in Experiment 7A, broken down by whether the targets were in- or out-of-subset during the previous blocks.

Subjective responses

User response to StencilMaps was overwhelmingly positive, with StencilMaps scoring higher on all statistically significant NASA-TLX measures (Table 7.1).

When asked which interface they preferred, 16 of 18 participants chose StencilMaps, one chose Ephemeral, and one did not answer, supporting **H3** ($\chi^2 = 28.4$, $p < 0.0001$.

7.2.6 Summary

The results of Experiment 7A are summarised below, according to the hypotheses and expectations:

H1: StencilMap will be faster than both Ephemeral and CommandMap for in-subset selections. *Supported.* Both Ephemeral and StencilMap were faster than CommandMap when it came to selection of items in the subsets. Further, out of the two subset presentation techniques, StencilMap

	CommandMap	Ephemeral	StencilMap	χ_r^2	Significance (p)
Mental demand	**4.0 (1.0)**	**3.5 (0.7)**	**2.7 (1.1)**	**11.7**	**0.002**
Physical demand	**3.2 (1.6)**	**2.7 (1.2)**	**2.2 (1.2)**	**8.0**	**0.018**
Temporal demand	3.5 (1.0)	3.3 (1.1)	2.9 (1.4)	3.4	0.186
Success	**3.5 (0.7)**	**3.9 (0.7)**	**4.1 (0.8)**	**7.2**	**0.027**
Hard work	**4.4 (0.9)**	**3.4 (1.0)**	**3.1 (0.9)**	**14.8**	**0.001**
Frustration	**3.3 (1.0)**	**2.4 (0.9)**	**1.8 (0.7)**	**14.8**	**0.001**

Table 7.1: Mean (s.d.) NASA-TLX responses for Experiment 7A according to a 5-point Likert scale (1 = low, 5 = high). Statistically significant measures are shown in bold.

was faster than Ephemeral, with performance advantages persisting until the 5th selection of each item.

H2: Overall mean selection times for StencilMap will be faster than Ephemeral and CommandMap. *Supported.* When considering both selections within the subset and outside of the subset, participants were significantly faster with StencilMap than the other two interfaces.

H3: Users will subjectively prefer StencilMap over Ephemeral and CommandMap. *Supported.* Preference and workload data were both strongly in favor of the StencilMap design.

E1: Overall mean selection times for Ephemeral will be faster than CommandMap. *Supported.* Ephemeral highlighting provided a performance benefit over the base CommandMap, although not as much as the StencilMap.

E2: CommandMap will perform better than both Ephemeral and StencilMap for out-of-subset items. *Supported.* For out-of-subset selections there is a trend suggesting advantages for the CommandMap. In other words, when users are not able to make use of a subset, they are better off with the full, unmodified CommandMap; however, the performance advantages for items that can be selected from within the subset outweigh this cost.

E3: Performance on the recall test will be no different between the three conditions. *Not supported.* The results reveal performance advantages for

both the CommandMap and ephemeral highlighting over the StencilMap when it comes to learning the location of items in the full interface. This is an interesting result, addressed further in the Discussion section.

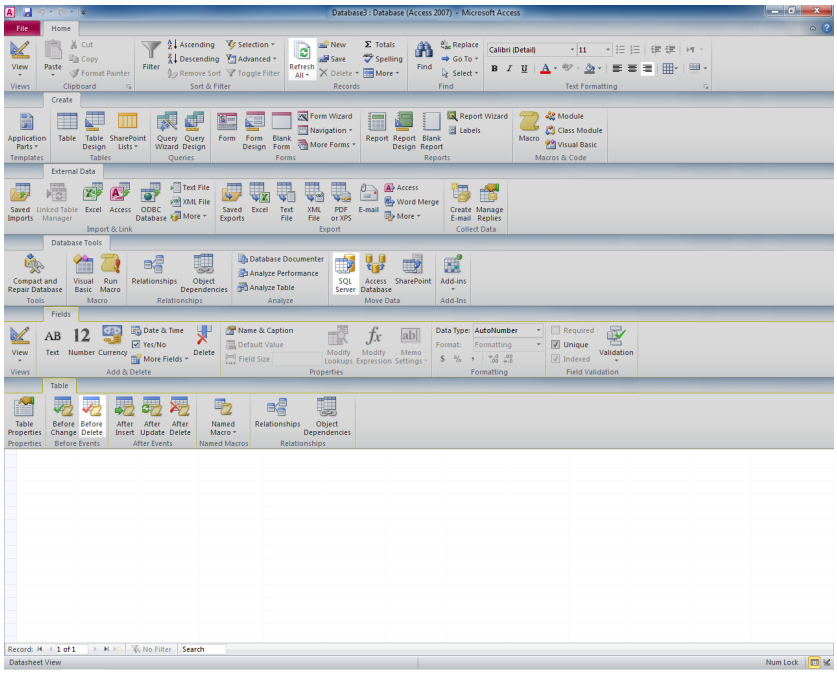
7.3 Experiment 7B: StencilMaps vs. palettes

A second user study was carried out to compare the performance of StencilMaps against the common subset design of using palettes of clustered items. The primary goal was to examine the trade-offs and differences between StencilMaps and palettes in terms of both speed of selection from subsets, and long-term item location learning. A secondary goal was to examine StencilMap performance in interfaces with realistic icon sets.

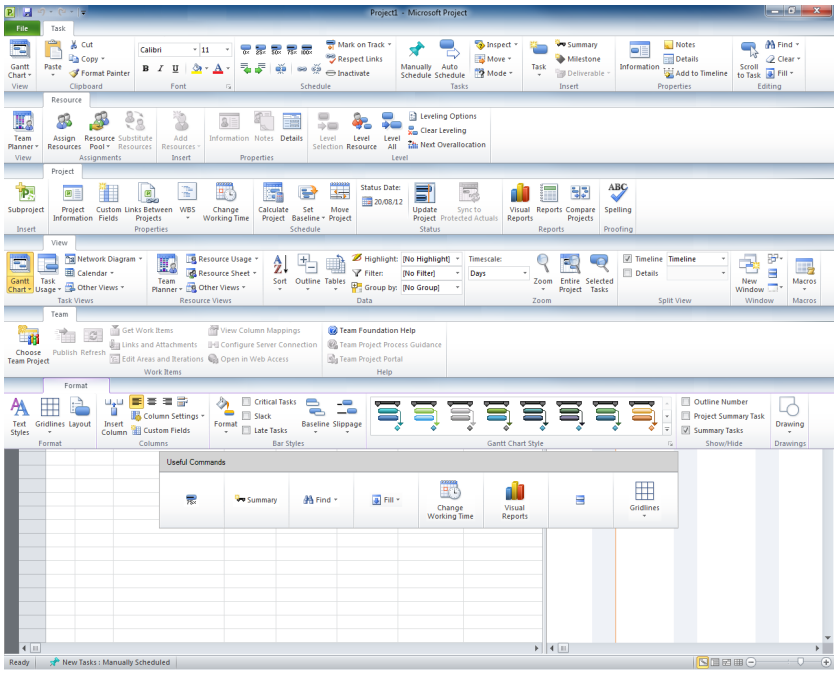
7.3.1 Procedure

Participants first completed a demographics questionnaire and were introduced to the study system. Two interfaces were evaluated: a StencilMap, and a CommandMap with a separate palette labeled "Useful Commands". A CommandMap was chosen, rather than a menu or ribbon interface, as the full interface for the Palette condition in order to have a fair comparison with StencilMaps: menus and Ribbons are known to be mechanically slower than CommandMaps for familiar items (see Chapter 5), so it would have been impossible to draw conclusions about learning effects based on selection time alone.

The CommandMaps were generated from the interfaces of two real-world applications: Microsoft Access (Figure 7.8a) and Microsoft Project (Figure 7.8b). These were chosen in order to provide a realistic icon set and semantic layout, while still making it easy to find participants who had not used the interfaces before. The Microsoft Access interface was always displayed first, but the order of the StencilMap and Palette was counterbalanced between participants. The study was composed of repeated cued selection tasks, using an experimental system analogous to Experiment 7A. In the StencilMap condition, participants had to select the item from the augmented CommandMap; in the Palette condition, participants could choose to use either the CommandMap or the always-visible palette (if it contained the target item). Since targets in the interface used a mixture of iconic and textual labels, both the icon and the name of the command were



(a) A 4-item Microsoft Access StencilMap.



(b) A Microsoft Project CommandMap with an 8-item palette located below the full CommandMap.

Figure 7.8: The interfaces used in Experiment 7B.

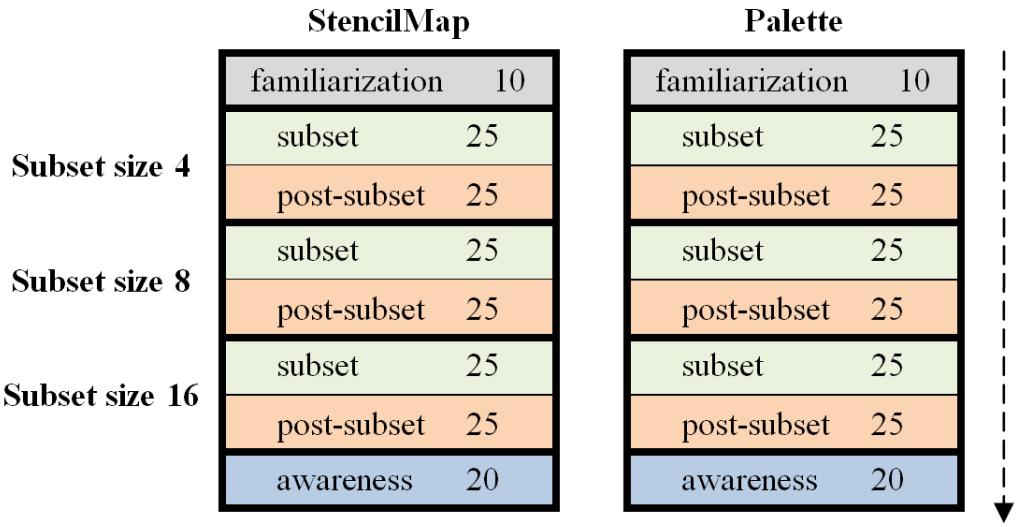


Figure 7.9: The experimental procedure used in Experiment 7B. Ordering of the Subset Size and Interface factors were counterbalanced. In total, each participant performed 360 trials.

used as the stimulus.

The experiment was again divided into three phases: *familiarization*, *selection*, and *awareness*. With each interface, participants first performed a *familiarization* block of 10 trials (data discarded), introducing them to the interface and experimental procedure. Participants then entered the *selection* phase, where they interacted with subsets of three different sizes: 4, 8, and 16 items. Each subset size condition consisted of two blocks: *subset* and *post-subset* (see Figure 7.9). Similar to Experiment 7A, the *subset* block comprised 25 selections from a subset interface (five selections of each of five targets), some of which were within the subset, and some of which were not (see Accuracy section below). The *post-subset* block consisted of the same sequence of items but with the subset removed, leaving participants with the full, unmodified CommandMap. More specifically, for the StencilMap interface, the post-subset block removed the stencil overlay, whereas in the Palette interface the separate palette was removed. The post-subset block was designed to analyse learning effects, similar to the recall block in Experiment 7A, with the differences being that the recall block in Experiment 7A took place at the end of the study and involved only a single selection of each item.

After finishing subset and post-subset blocks with all three subset sizes, par-

ticipants performed an *awareness* test using Findlater and McGrenere's measure of 'indirect awareness' [55]: participants were asked to perform 20 further selections of items that were not targets in the main experiment. The awareness test measures incidental learning of items not part of the main experiment (and therefore differs from the recall test in Experiment 7A, which examined recall times of selections that were made previously with the subsets present).

The complete experimental procedure is displayed in Figure 7.9. The target sets, selection sequences, and subsets were randomly generated for each participant.

7.3.2 Accuracy

While Experiment 7A focused on a subset accuracy level (80%) where highlighting is known to be beneficial, Experiment 7B also investigates the performance of each technique when subset accuracy (i.e. the percentage of required items that are present in the current subset) is low. Manipulating this factor enables a better understanding of the differences between palettes and StencilMaps as the proportion of commands in the subset varies. Accuracy was therefore manipulated as a between-subjects factor: 12 participants were exposed to a Low accuracy level of 40%, while the other 12 were exposed to a High accuracy of 80%. These levels are based on similar values used by Findlater et al. [57].

7.3.3 Participants and apparatus

24 participants were recruited from a local university: 8 male, 16 female, mean age 25 years (s.d. 4.2). Hardware and setup was the same as Experiment 7A.

7.3.4 Design and hypotheses

Selection time is analysed using a 2×3×2×2 mixed-design ANOVA for within-subjects factors Interface (*stencilmap*, *palette*), Subset Size (*4*, *8*, or *16* items), and Block (*subset*, *post-subset*); and between-subjects factor Accuracy (*high*, *low*). Order of Interface and Subset Size was balanced using a Latin square.

Formal hypotheses for the experiment were as follows:

H1: StencilMaps will be faster than palettes for post-subset trials. Since

StencilMaps do not change item locations, participants' location learning during the subset phase should allow them to perform faster in the post-subset phase than with palettes.

H2: Palettes will be faster than StencilMaps for subset trials. The advantage of a palette is that it displays promoted commands in a compact, easy-to-search arrangement. Palettes are therefore expected to be faster than StencilMaps during the subset phase.

H3: StencilMaps will be faster than palettes overall. The performance advantage of StencilMaps in the post-subset phase should outweigh the in-subset performance advantage of palettes.

H4: StencilMaps will be faster than palettes for the awareness test. Since the StencilMap requires users to interact with the main interface, participants should develop better overall command awareness than with the palette.

7.3.5 Results

Error rates were low: 4.4% for *stencilmap*, and 5.4% for *palette* (no significant difference). The following analyses exclude trials with errors.

Overall selection times

Figure 7.10 summarizes the results. Overall, *stencilmap* (2.64s, s.d. 1.29) was faster than *palette* (2.90s, s.d. 1.13), giving a significant main effect of Interface ($F_{1,22} = 6.5$, $p = 0.018$, $\eta^2 = 0.23$). This result supports **H3**. There was also a significant main effect of Block ($F_{1,22} = 40.6$, $p < 0.001$, $\eta^2 = 0.65$), with *post-subset* (2.38s, s.d. 0.87) faster than *subset* (3.16s, s.d. 1.39).

Four interaction effects were significant. Importantly, there was a significant effect of Interface×Block, ($F_{1,22} = 34.1$, $p < 0.001$, $\eta^2 = 0.61$), with *stencilmap* showing a large advantage on post-subset selections (Figure 7.10). Planned pairwise comparisons confirmed that *stencilmap* was faster than *palette* in *post-subset* blocks ($p < 0.001$), and that *palette* was faster than *stencilmap* in *subset* blocks ($p = 0.03$). These results provide support for **H1** and **H2**.

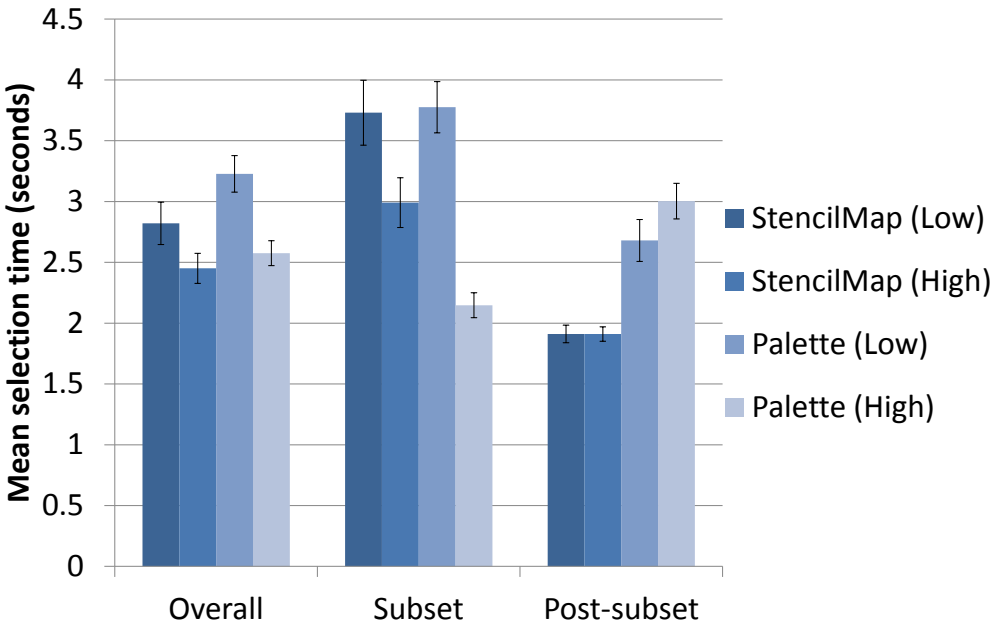


Figure 7.10: Mean selection times for StencilMaps and palettes for subset and post-subset blocks, across *low* and *high* accuracy.

There were also significant interaction effects for Accuracy×Block ($F_{1,22} = 34.1$, $p < 0.001$, $\eta^2 = 0.58$), Interface×Accuracy×Block ($F_{1,22} = 7.1$, $p = 0.014$, $\eta^2 = 0.24$), and Subset Size×Block ($F_{2,44} = 5.1$, $p = 0.011$, $\eta^2 = 0.19$). As shown in Figure 7.10, the main difference between the *palette* and *stencilmap* techniques for the *subset* block was at the *high* subset accuracy levels: at *low* accuracy, there was very little difference between the two techniques.

Figure 7.11 clearly displays the difference in item location learning between StencilMaps and palettes. While participants were 2.3s slower with StencilMaps on the first selection of each item in the subset block, they performed much faster in the post-subset block (more than 4s over the first two selections).

Palette use

In the subset blocks for the *palette* condition, participants had the choice to use either the palette or the full interface. For selections where the item was present in the subset, participants chose to use the palette 95% of the time in the low accuracy

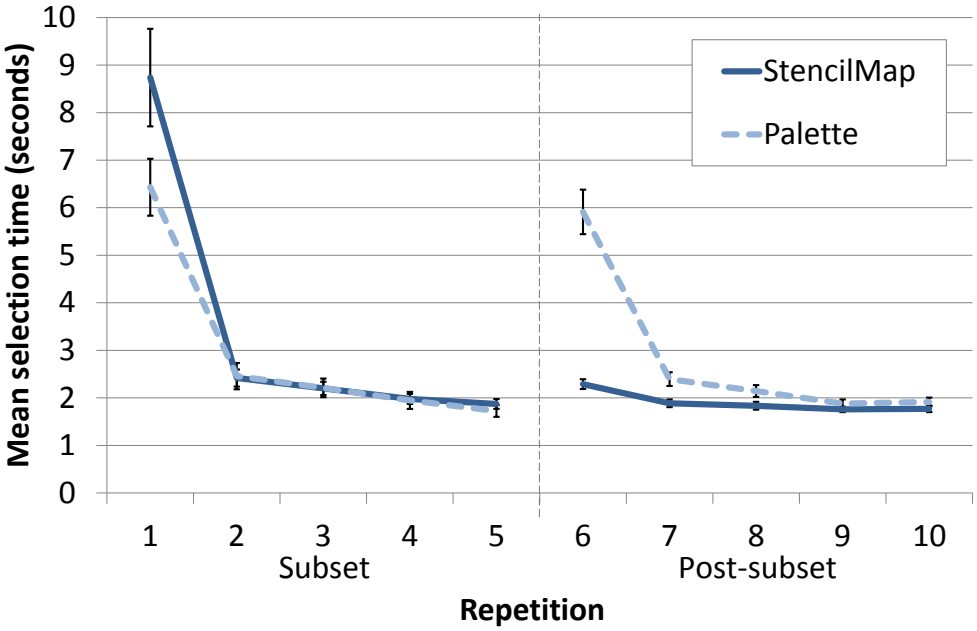


Figure 7.11: Mean selection times across item repetition.

condition, and 99% of the time in the high accuracy condition. This difference was marginally significant ($p = 0.04$). This shows that participants almost always searched the palette before the main interface, although this behavior decreased slightly when target items were in the palette less often.

Awareness

For the awareness test, a 2×2 ANOVA of Interface and Accuracy on selection time showed no significant effects for either Interface ($F_{1,22} = 1.0$, $p = 0.32$) or Accuracy ($F_{1,22} = 0.2$, $p = 0.64$). **H4** is therefore not supported.

Subjective responses

As shown in Table 7.2, there were no significant differences in NASA-TLX ratings between the two interfaces. When asked which interface they preferred, 14 participants chose StencilMaps, and 10 chose Palettes, also with no significant difference ($\chi^2 = 0.38$, $p = 0.54$).

	StencilMap	Palette	Wilcoxon z	Significance (p)
Mental demand	2.9 (1.1)	3.0 (1.3)	0.18	0.428
Physical demand	2.6 (1.3)	2.4 (1.3)	0.93	0.176
Temporal demand	2.9 (1.2)	2.8 (1.0)	0.20	0.422
Success	4.0 (0.9)	3.8 (0.8)	1.29	0.098
Hard work	3.2 (1.1)	3.1 (1.1)	0.21	0.416
Frustration	2.5 (1.1)	2.8 (1.0)	0.68	0.247

Table 7.2: Mean (s.d.) NASA-TLX responses for Experiment 7B according to a 5-point Likert scale (1 = low, 5 = high).

7.3.6 Summary

The results of Experiment 7B are summarised below, according to the hypotheses:

H1: StencilMaps will be faster than palettes for post-subset trials. *Supported.* The StencilMap supported the transition to the full interface much better than the palette interface.

H2: Palettes will be faster than StencilMaps for subset trials. *Supported.* Participants were able to leverage the compact representation of the palette to make faster selections when the subset was present than was the case with the StencilMap.

H3: StencilMaps will be faster than palettes overall. *Supported.* Participants were faster with the StencilMap than with the palette when considering selections made both when the subsets were present and when they were removed.

H4: StencilMaps will be faster than palettes for the awareness test. *Not supported.* There was no difference in incidental learning between the two subset presentation techniques.

7.4 Discussion

To summarise the results of both experiments, Experiment 7A showed that users are able to locate unfamiliar items more quickly with highlighting, and that StencilMaps are more effective at reducing search time than ephemeral highlighting.

Experiment 7B showed that StencilMaps allow for better long-term location learning in the full interface compared to a palette-based subset approach, with the performance benefits after switching to the full interface outweighing the small performance penalty in the subsets.

However, StencilMaps caused a slight decrease in participants' ability to recall the locations of in-subset items they had seen in the Experiment 7A (Figure 7.7). The sections below discuss this issue, as well as the generalisability of the results of both experiments.

7.4.1 Why were recall selections slower after using the StencilMap?

StencilMaps are designed with two goals in mind: faster selection of subset items by novices, and improved spatial location learning compared to other subset methods. The negative results for StencilMaps in the *recall* phase of Experiment 7A are therefore a concern.

This result is in line with the *guidance hypothesis* of skill acquisition [175]. The guidance hypothesis suggests that providing extra guidance to improve early task performance may impair retention of the performed skill once the guidance is removed. One explanation for this effect is given by Craik and Lockhart's 'levels of processing' framework [38], which argues that 'deeper', more involved cognitions result in better memory retention than shallower cognitions (see Section 2.3.1). With respect to this hypothesis and framework, it is plausible that the effective highlighting provided by StencilMaps induced a more "shallow" encoding of item locations because the in-subset items were almost immediately ready to hand and identifiable, whereas items in the CommandMap required extensive visual search, prompting participants to think more carefully about target item locations.

Interestingly, this supports the idea presented in Section 3.3 that there may be an inherent trade-off between making things easier for novices and supporting long-term application learning. It is worth noting, however, that the practical difference between the two conditions was low (only 0.78s), suggesting that StencilMaps provide a good compromise between the two goals. Furthermore, participants made only one selection of each item; based on the learning curves shown in Figures 7.6 and 7.11, it is likely that this difference would have been even lower

after multiple selections.

In general, there are extensive opportunities for further work in examining how assistive highlighting may be gradually tailored to best facilitate the dual purposes of efficient novice interaction and early attainment of expertise (e.g., [5]).

7.4.2 Generalisation and limitations

While StencilMaps performed favorably in both experiments, there are limitations to the experiments that could affect the generality of the results. These limitations are discussed below.

Sensitivity to onset time

As discussed prior to Experiment 7A, the performance of ephemeral highlighting is sensitive to onset time. There is therefore a risk that the onset times used (500ms, 1000ms, and 2000ms) were not optimal for the experimental task: they may have been too short, not allowing enough time for participants to search the visual subset; or too long, unnecessarily delaying participants in finding non-predicted items. However, mean selection times in the ephemeral condition were higher than StencilMaps for both in-subset and out-of-subset selections. Since adjusting ephemeral onset time is necessarily a performance trade-off between in- and out-of-subset items, changing the onset time in either direction would not have resulted in an overall win for ephemeral. Furthermore, the need to tweak onset time is an inherent weakness of the ephemeral technique: the optimal onset time is dependent on the size and distribution of the subset, as well as the abilities of the user. Stencils, which are permanent, avoid this problem, easing real-world implementation.

Source of subset commands

Command subsets can be populated from many different sources. For example, an adaptive system dynamically builds subsets based on predictions about the user's upcoming actions; and palettes in real-world interfaces can often be personalized to match the user's evolving needs.

However, in the two experiments described here, only static, pre-defined subsets were considered. This was done for two reasons. First, the primary goal of

StencilMaps is to assist novice users in gaining expertise, and in this context, pre-populated 'training wheels' interfaces have shown promising results [23]. The intended use case for StencilMaps is for subsets to be populated with relevant commands for specific tasks or workflows, similar to those used in AdaptableGIMP. Novice users can turn on the stencil as a complexity-reducing tool while they learn how to perform a new task; they can then turn off the stencil once it is no longer needed, keeping their newly-found spatial knowledge intact. Second, pilot studies revealed that subset interfaces have many subtle attributes that can affect performance – the goal in this work was therefore to test the simplest possible case.

7.4.3 Stencils in other interfaces

One issue to be considered in deploying the stencil approach in real-world systems is whether the technique can work with interfaces that do not make the entire subset visible – for example, some commands may be hidden behind menus or ribbons. Stencil subsets can still be used in these interfaces: the stencil simply highlights the titles of any menus or ribbons that contain items of the subset, and the user must open the hierarchical control to find those items. One example is Fitchett et al.'s Search-Directed Navigation interface [59], which uses stencils to highlight items in a file hierarchy.

7.5 Conclusions

Subset interfaces can make it easier for novices to interact with complex interfaces. However, subset interfaces that spatially rearrange controls can hinder novices' transition to the full UI, because the user must learn new command locations in the full interface. To address this problem, this chapter presented the StencilMap, which combines a spatially consistent CommandMap with a stencil overlay to visually highlight the subset.

An initial experiment evaluated the effects of stencil overlays on visual search in CommandMaps, comparing it to ephemeral highlighting. Results indicated that the non-transient highlighting of StencilMaps gave it an advantage over ephemeral highlighting for in-subset selections, with no significant difference for out-of-subset selections. The experiment did, however, reveal a performance trade-off

between the two highlighting techniques, with StencilMaps providing better support for visual search, and ephemeral highlighting leading to better location learning. Of the two designs, users expressed a strong preference for StencilMaps.

Experiment 7B compared the performance of StencilMaps to palette-based subsets in a realistic Microsoft Office interface. The results show that users were slightly faster with palettes than StencilMaps in the subsets, but that StencilMaps led to much faster performance once the subset was removed. Subjective responses indicated that StencilMaps were seen as equivalent to palettes by participants.

The results from the two studies combined indicate that stencil overlays provide a promising mechanism for aiding novice interaction with CommandMaps, and that they warrant further study. A summary of promising future research directions on StencilMaps is provided in the thesis discussion (Chapter 9).

Part III

Supporting Spatial Memory Under Changing Interface Constraints

Chapter 8

Maintaining Spatial Consistency when Display Bounds Change

Both CommandMaps (Chapters 5 and 6) and StencilMaps (Chapter 7) are maximally flattened interfaces that utilise as much of the screen as possible. The success of these types of interfaces relies heavily on spatial memory, which allows users to find controls quickly assuming that they stay in constant locations.

However, there are situations that arise with any spatially-organised interface where it is difficult or impossible to maintain the spatial positions of items. In particular, changing the display geometry (such as resizing a file browser window, or rotating a smartphone display) means that the interface must somehow adapt to the new display bounds. In these situations, interface designers can choose how the interface reacts.

Most often, current interfaces adapt to changing window geometries in ways that interfere with the use of spatial memory. For example, when the user changes the size of a window, or rotates a tablet computer from landscape to portrait mode, many systems reflow items from left-to-right, top-to-bottom within the window bounds – that is, they re-arrange items to fit the new aspect ratio of the window or display (Figure 8.1). This fills the window, but because item locations have changed, they can be more difficult to find. Similarly, in command interfaces like the Ribbon, controls are repositioned, rearranged, and elided into dropdown menus (Figure 5.15).

This chapter investigates an alternative method of adapting to changing window bounds, utilising the idea of *frames of reference* (see Section 2.4.1). Since people typically learn spatial locations in terms of a frame of reference, this chapter proposes the use of *spatially consistent transformations* to maintain spatial memory when display bounds change.

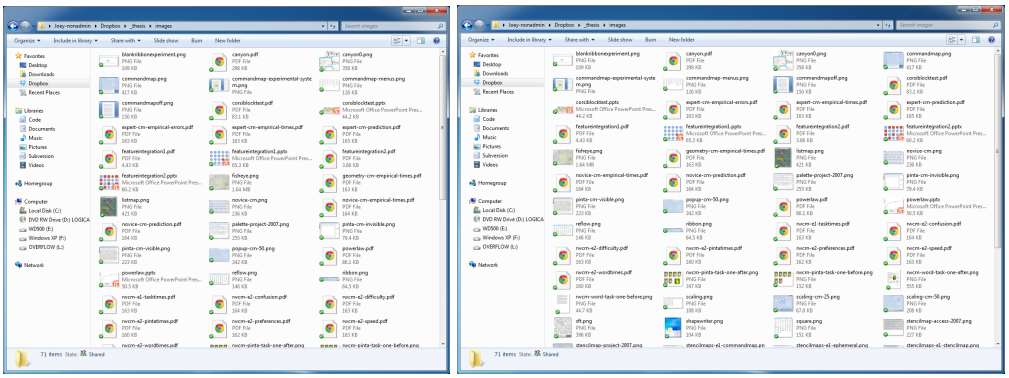


Figure 8.1: Windows Explorer in tile view, before and after a change in window dimensions.

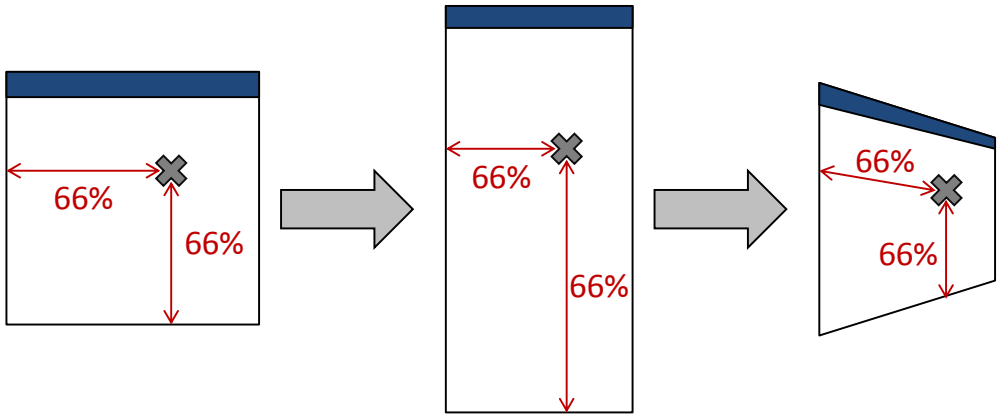


Figure 8.2: Spatial consistency keeps items proportionately stable with respect to the window bounds.

A spatially consistent transformation is defined in this chapter as a geometric transformation that maintains item locations relative to some frame of reference, such as the window bounds (Figure 8.2). For example, if an item is in the top-right corner of the of window before transformation, it should still be in the top-right corner after transformation.

This chapter presents a human-factors experiment examining the effects on spatial memory of various types of spatially consistent transformation: translation, scaling, stretching, rotation, and perspective change. The results of this experiment support the formation of a new design guideline advocating *spatial*

consistency relative to a frame of reference. This guideline is used to design a spatially consistent file browser, which is compared in a second experiment to the standard technique of reflowing items when the window size changes.

Although the techniques presented in this chapter are applied to file browsers, they also apply to other spatially-organised interfaces, such as CommandMaps and StencilMaps.

8.1 Spatial consistency as a design guideline

'Be consistent' is a fundamental rule of HCI, featuring in many design guidelines (e.g. [2, 154, 181]). However, this principle is abstract, and it does not prescribe which design elements should be held consistent. Hansen's 1971 interface guidelines [79] include a recommendation to support 'display inertia', meaning that "the size and layout of the display do not change drastically" (p. 529). Hansen's objective in this guideline was to optimize user execution of operations by allowing users to make rapid decisions (modeled by Hick-Hyman choice reaction time [86, 96]).

However, Hansen's argument that the size of the display should not change drastically is inconsistent with current interface designs, where users have freedom to resize and reorient windows. This chapter investigates methods to achieve display inertia that are robust to commonly occurring size and layout manipulations.

In this chapter, the design principle 'maintain relative spatial consistency within the frame of reference' is proposed as a foundation of interface organization. The frame of reference will normally be provided by the display edge or by the borders on a particular window, but it can also be perceived by Gestalt proximity [208]: for example, a grid of items with no visible border can still be seen to have a frame of reference. 'Relative spatial consistency', means that the arrangement of items within the frame of reference should remain proportionately stable with respect to the bounds of the frame. For example, if an item is the closest item to the top right corner of a frame before transformation, it should be similarly positioned after transformation as well. Figure 8.2 illustrates relative spatial consistency as the frame undergoes stretch and perspective transformations.

8.1.1 Common transformations to the frame of reference

Frames of reference in UIs commonly undergo (or are viewed in such a way that they are perceived to undergo) five forms of visual transformation (Figure 8.3). The thick 'top' edges of the frames in Figure 8.3 represent the standard orientation of the frame (i.e., which way is 'up').

1. **Translation** occurs frequently in desktop computing, when windows are moved to different screen locations.

2. **Scaling** also occurs frequently in desktop computing, when windows are resized by the user.

3. **Stretching** (changing aspect ratio). Similar to scaling, stretching occurs when windows are resized in one dimension. This also occurs on mobile devices when an interface is reoriented to landscape or portrait mode.

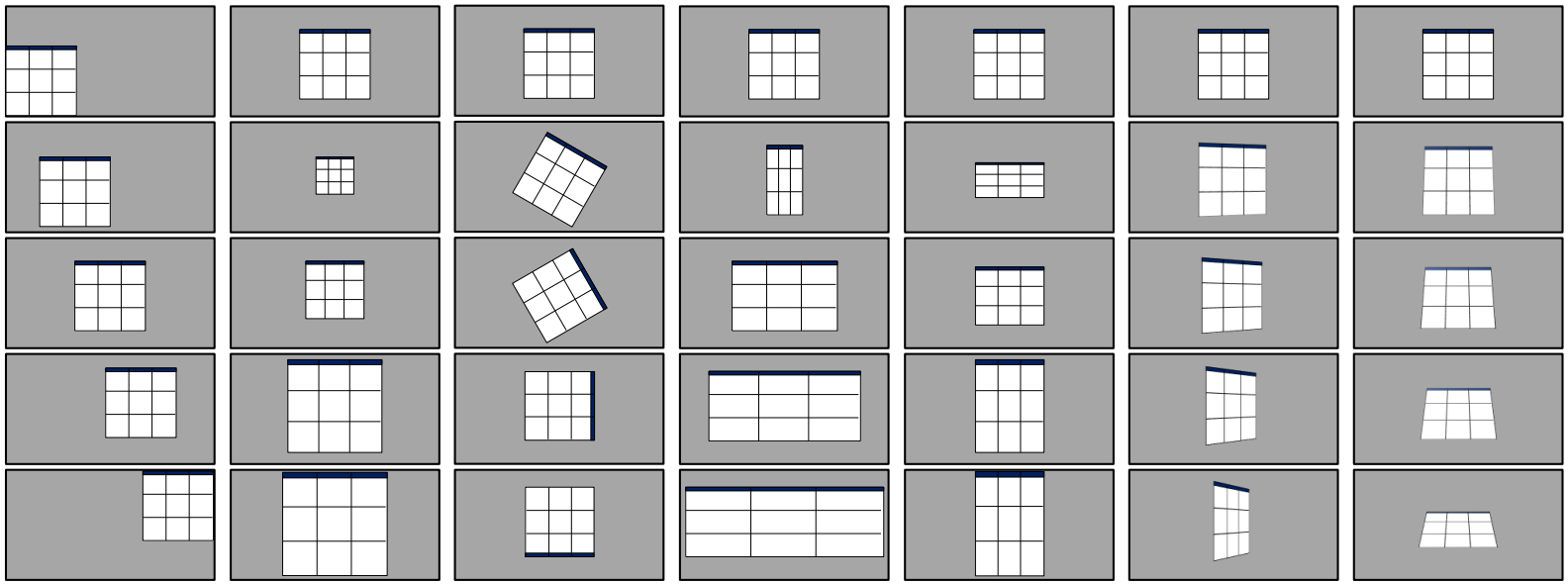
4. **Rotation** is common in surface-based computing (e.g. digital tables or shared use of tablets), where displays or windows can be turned to face another person. It also occurs on mobile devices when an interface has not been programmed to adapt to device rotation (e.g., the Apple iPhone home screen, when viewed in landscape mode).

5. **Perspective distortion** occurs when viewing any kind of display from an oblique angle, as is common on shared wall or tabletop displays.

8.1.2 Relative spatial consistency after transformations

The previous section discussed transformations to the frame of reference itself. Relative spatial consistency, however, concerns the location of content inside the frame of reference after transformation.

When the frame of reference changes, UI designers can choose how the interface adapts to the new bounds. Translation, rotation and perspective transformations normally do not require any particular adaptation or response from the user interface – the window moves (with translation) or the user changes their viewing orientation (with rotation and perspective). However, an interface response is



(a) Translation (b) Scaling (c) Rotation (d) Stretching (horizontal) (e) Stretching (vertical) (f) Perspective (horizontal) (g) Perspective (vertical)

Figure 8.3: The transformations used in Experiment 8A. The top row shows the untransformed window used for training.

$$\begin{pmatrix} 1 & 0 & \Delta x \\ 0 & 1 & \Delta y \\ 0 & 0 & 1 \end{pmatrix}$$

(a) Translation

$$\begin{pmatrix} k & 0 & 0 \\ 0 & k & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

(b) Scaling

$$\begin{pmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

(c) Rotation

$$\begin{pmatrix} k & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

(d) Stretching (horizontal)

$$\begin{pmatrix} 1 & 0 & 0 \\ 0 & k & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

(e) Stretching (vertical)

$$\begin{pmatrix} \cos\theta\sqrt{\tan^2\theta+1} & 0 & 0 \\ 0 & \sqrt{\tan^2\theta+1} & 0 \\ \sin\theta & 0 & \cos\theta+\sin\theta\tan\theta \end{pmatrix}$$

(f) Perspective (horizontal)

$$\begin{pmatrix} \sqrt{\tan^2\theta+1} & 0 & 0 \\ 0 & \cos\theta\sqrt{\tan^2\theta+1} & 0 \\ 0 & \sin\theta & \cos\theta+\sin\theta\tan\theta \end{pmatrix}$$

(g) Perspective (vertical)

Figure 8.4: Transformation matrices for the seven geometric transformations used in Experiment 8A.

necessary when the user scales or stretches the frame of reference. 'Reflowing' the content is a common design strategy (e.g., the grid view in the Windows file explorer), as is item elision (e.g., the Office 2007 Ribbon moves items into hierarchies as the window gets smaller). As demonstrated by Experiment 8B, designers could also choose to maintain the original arrangement of items within the frame, and scale the entire grid when a stretch occurs.

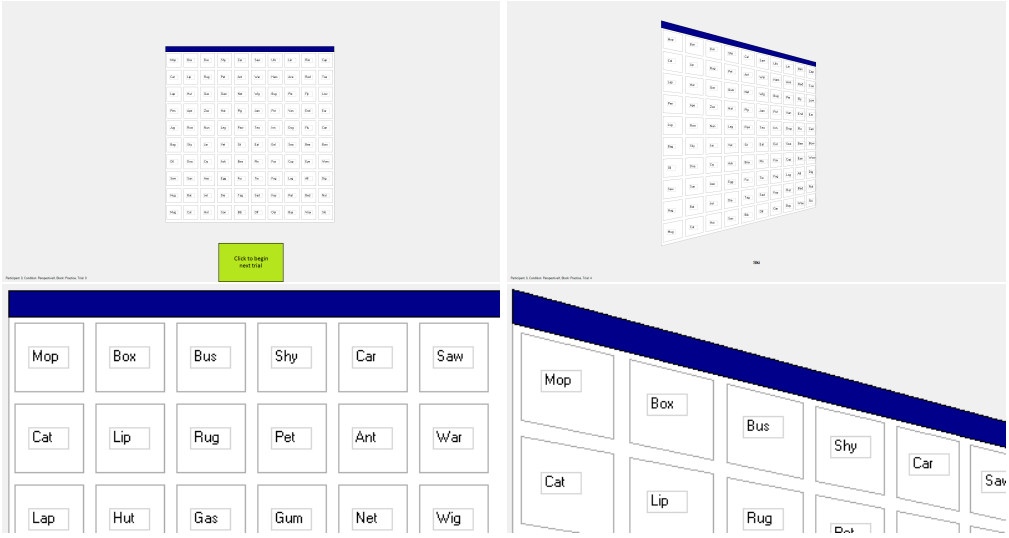
The grid lines in each window in Figure 8.3 depict how 'canonical' relative spatial consistency can be maintained during the different transformations. Other approaches (e.g., 2D scaling in response to 1D stretching) can be achieved by combining these primitives. Figure 8.4 shows transformation matrices for each effect.

8.2 Experiment 8A: Studying interface transformations

It seems reasonable to assume that the fast interaction enabled by spatial consistency will be robust to at least some of the transformations described above and shown in Figure 8.3. For example, users are unlikely to have difficulty locating items in a window after translating it. However, the time taken to adapt and respond to these transformations is less clear – while it is likely that users can reorient their spatial understanding, there may be time costs in doing so. Lam et al.'s experiment on 2D geometric transformations [119] provides some insight, showing that performance is not affected within certain "windows" for each type of transformation; however, their experiment studies recognition of shapes, rather than selection from a grid of items.

Experiment 8A, therefore, was an exploratory study designed to determine the additional time needed to find items following different types and magnitudes of spatially consistent transformation. The method involved repeatedly selecting the same four items in a spatially consistent layout, while the grid underwent the transformations shown in Figure 8.3.

The time taken to select a target after each transformation involves perceptual and cognitive processes of reorienting to the display and deciding about item location (labelled here as 'reorientation time'), as well as the mechanical time to point to the target. The variable of interest in this experiment is reorientation time, so to extract the variable effects of target relocation caused by the transformations,



(a) Untransformed (b) Transformed

Figure 8.5: Overview and close-up of the interface used in Experiment 8A, before and after a horizontal perspective transformation.

each participant's Fitts' Law pointing characteristics were analysed and used to subtract pointing time from the total selection time for each item. (Note that Experiment 8B considers overall performance including pointing time; the goal of Experiment 8A is to study the reorientation phase.)

8.2.1 Tasks, stimuli, and instructions

Tasks involved a sequence of selections from a 10×10 grid of textual items (Figure 8.5a) that was either transformed or untransformed. All text items were common English three-letter words, which were used (rather than images or variable word sizes) to reduce confounds from visual pop-out. Also, to avoid effects of reading distorted text, text labels were not transformed with the interface (Figure 8.5b). An exception was made for rotation because pilot testing showed that participants use text orientation as a primary cue to establishing the frame of reference.

Tasks began by showing an untransformed window (Figure 8.5a). Participants clicked a button to reveal the target item and display the transformed window (Figure 8.5b). Task timing began with the button click, and stopped when the target was selected; this was achieved by clicking in a visible hitbox surrounding the

Transformation	**Magnitude**			
	1	2	3	4
Translation $(\Delta x, \Delta y)$	290px, 100px	580px, 200px	870px, 300px	**1160px, 400px**
Scaling	$\times$**0.5**	$\times$0.75	$\times$1.25	$\times$1.5
Rotation	30°	60°	90°	**180°**
Stretching (horizontal)	$\times$**0.5**	$\times$1.5	$\times$2	$\times$2.5
Stretching (vertical)	$\times$**0.5**	$\times$0.75	$\times$1.25	$\times$1.5
Perspective (horizontal)	15°	30°	45°	**60°**
Perspective (vertical)	15°	30°	45°	**60°**

Table 8.1: Experiment 8A transformations and magnitudes. Bold items denote the level deemed most extreme.

text label, which was a constant size in all conditions. After selection, the display returned to the untransformed window. Subjects were asked to make selections "as quickly and accurately as possible".

8.2.2 Transformations and magnitudes

The study tested seven transformations: translation, scaling, rotation, horizontal stretching, vertical stretching, horizontal perspective, and vertical perspective (columns in Figure 8.3). Each transformation was tested at five levels of magnitude (rows in Figure 8.3). A summary is shown in Table 8.1.

8.2.3 Procedure

Each participant initially performed a bi-directional Fitts' calibration task, consisting of 144 selections across 7 indices of difficulty. They then completed four blocks of trials with each of the seven transformations (Table 8.1). Order of transformation was counterbalanced using a Latin square. All four blocks were completed with one transformation before advancing to the next. The blocks were *familiarisation*, *training*, *recall*, and *learning*, always in that order.

The *familiarisation* block (data discarded) acquainted participants with the transformation, and consisted of ten trials (two for each magnitude), using different target items to the main experiment.

The *training* block consisted of 20 trials in the untransformed interface. The *training*, *recall*, and *learning* blocks used the same four target items through-

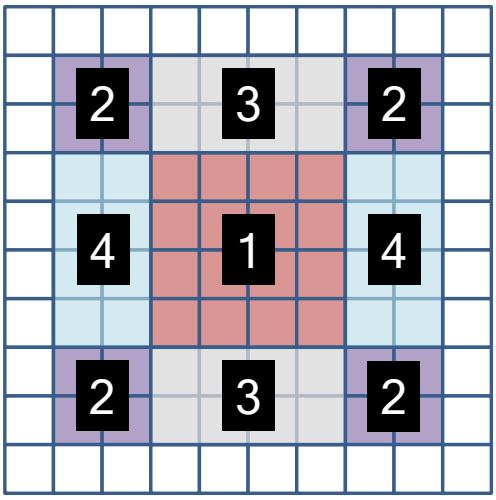


Figure 8.6: The four areas of the 10×10 grid from which target items were randomly selected for Experiment 8A.

out the experiment. To reduce potential confounds stemming from specific item locations, each participant had a unique set of target locations, with each item randomly selected from one of the four regions shown in Figure 8.6. No adjacent locations were allowed.

The recall block was used to examine selection times immediately after transformation. It consisted of 20 selections: one each for the four target items at each of the five magnitudes of transformation, in random order. The un-transformed interface was displayed between trials, and became transformed once the participant initiated the trial.

Finally, the learning block was included to examine participants' ability to relearn item locations after the interface had been transformed. This block used the most extreme form of each transformation (bold items in Table 8.1), and participants selected each target five times (random order), without the untransformed window being presented between selections (i.e., the extreme view was continually shown).

In summary, each participant performed 1960 trials:
7 transformations × 4 blocks
familiarisation: 10 selections (data discarded)
training: 20 selections

recall: 20 selections (using same 4 targets as *training*)
learning: 20 selections (using same 4 targets as *training* and *recall*)

8.2.4 Participants and apparatus

There were 14 participants; 7 male, 7 female, aged 19-42 (mean 26.9). The experiment was performed on a Windows 7 PC with a 1920×1200 monitor. Participants performed Experiments 8A then 8A in a single one-hour session.

8.2.5 Design

The study compares the time needed to reorient to a transformed display to the time for the non-transformed view. Reorientation time (T_r) is calculated by subtracting pointing time (T_p) from total selection time (T). Pointing time is calculated using each participant's individually calibrated Fitts' Law function, so $T_r = T - T_p$. For each transformation type, two pairwise measures are used to characterise the size of the effect of each transformation magnitude in comparison to the non-transformed condition: the statistical effect size using Cohen's d, which provides a sample-size independent estimate of effect size (Cohen [36] states that .2 is a small effect, .5 is medium, and .8 large); and the percentage increase in reorientation time. Reorientation time is analysed using a 7×5 repeated measures ANOVA with within-subjects factors Transformation (*translation*, *scaling*, *rotation*, *stretchingX*, *stretchingY*, *perspectiveX*, *perspectiveY*) and Magnitude (*0*, *1*, *2*, *3*, *4*).

8.2.6 Results

Fitts' calibration

Linear regression showed strong Fitts' models for 13 participants ($R^2 \geq 0.95$), and one slightly weaker at $R^2 = 0.89$. The mean pointing time predicted by the models varied little between the transformation types and their magnitudes: the overall mean was 948ms (s.d. 25), ranging from 926ms in the 180° *rotation* condition to 1063ms in the maximum *translation* condition.

Time to find items during training

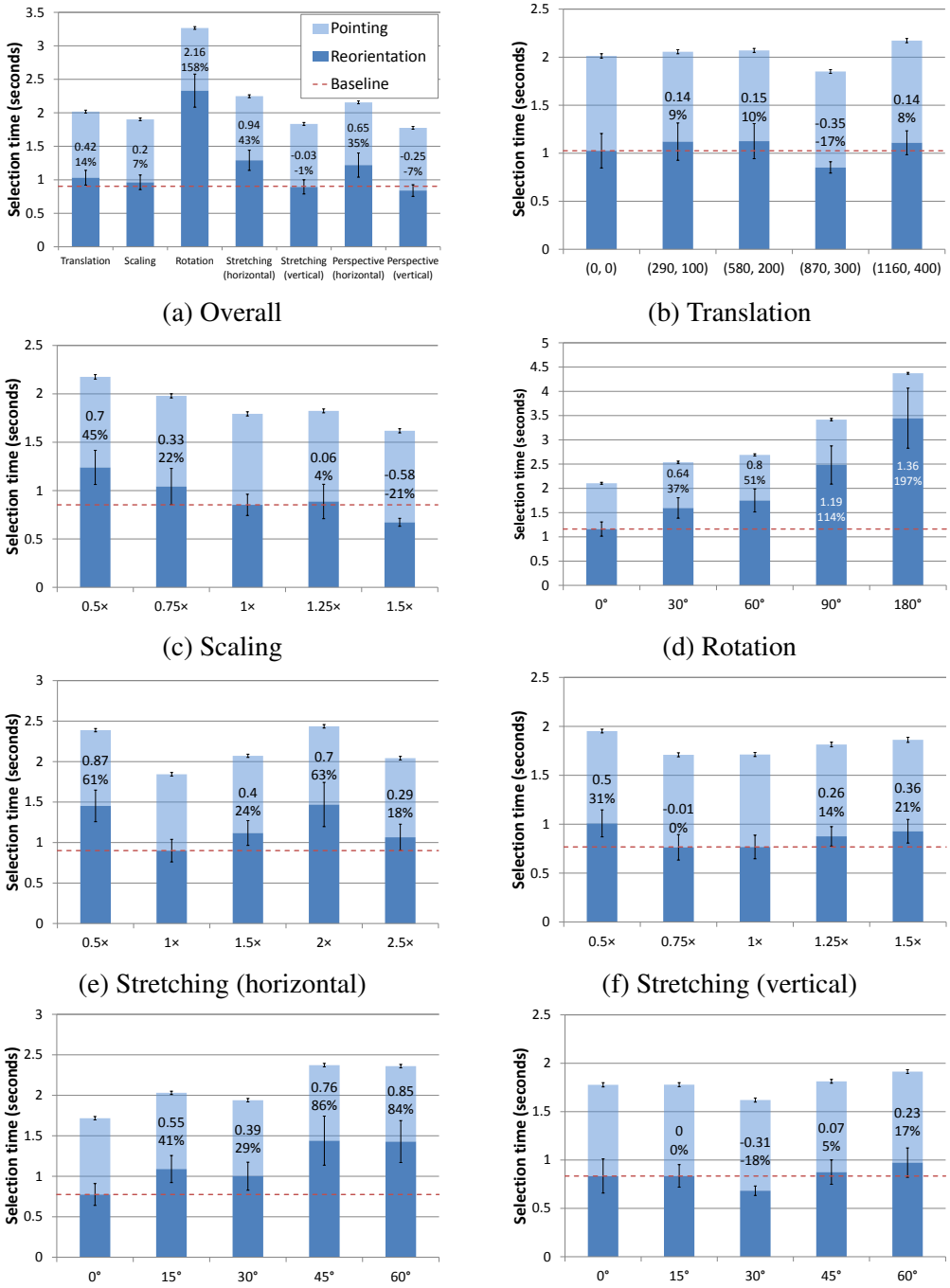
The experiment focuses on the additional time required for users to reorient to known spatially consistent displays when they undergo various forms of visual transformation. One relevant data point that helps understand the scale of reorientation cost (to determine whether the time increases are large or small) is the time taken to find the items when the user has no spatial knowledge regarding item placement. This section therefore analyses the mean time to select items for the first time in the training block, which occurred after familiarisation but before any spatial learning. The mean selection time was 15470ms, of which 14493ms can be attributed to visual search (once predicted pointing time is subtracted). By the fifth repetition during training, the mean decision time (selection time minus pointing time) had reduced to 811ms, which can be attributed to spatial memory supporting much faster selections.

Reorientation time after transformation in recall blocks

The primary results concerning reorientation times are presented here. Mean reorientation times across all levels for each transformation, as well as the mean calculated Fitts' Law pointing times, are shown in Figure 8.7a – the dashed horizontal line shows the mean reorientation time for the non-transformed condition. The lower segment of each bar shows reorientation time, and the upper segment shows calculated pointing time. The two numbers in each bar show Cohen's d effect size compared to the baseline and the percentage increase from the baseline. Figure 8.7a suggests that most of the transformations (other than rotation) had a relatively small impact on reorientation time – within 388ms of the baseline, which is only 2.7% of the visual search time reported above. As expected, ANOVA (error trials removed) showed significant main effects of Transformation ($F_{6,\,78} = 19.1,\ p < 0.001$) and Magnitude ($F_{4,\,52} = 8.5,\ p < 0.001$), as well as an interaction effect ($F_{24,\,312} = 4.8,\ p < 0.001$).

The analysis shows that adapting to transformed displays caused a reliable increase in reorientation time, but that this increase is small compared to the visual search time needed when the item's location is unknown.

Figure 8.7a shows reorientation time averaged across all transformation magnitudes (except the no-transformation level). To gain further insight into the



(a) Overall (b) Translation

(c) Scaling (d) Rotation

(e) Stretching (horizontal) (f) Stretching (vertical)

(g) Perspective (horizontal) (h) Perspective (vertical)

Figure 8.7: Reorientation time and Fitts' Law pointing times for each transformation type. The baseline value (red dashed line) is the mean reorientation time of participants in the un-transformed condition.

effects of each transformation magnitude, each transformation magnitude was individually compared with the no-transformation magnitude. The results are summarised in Figure 8.7 for each transformation, which include Cohen's d and percentage differences. Note that the baseline data is extracted from the no-transformation level within each transformation type. The key findings are as follows.

Translation caused small absolute increases in reorientation time (<100ms), regardless of magnitude (Figure 8.7b). This finding is unsurprising given users' extensive experience in adapting to windows placed in different display regions. The reduced time at the (870, 300) translation level is attributed to participants having already moved their mouse closer to this translation (which was near the centre of the screen) causing a reduction in actual pointing time, and hence an under-estimation in calculated reorientation time.

Scaling (Figure 8.7c) had little effect on reorientation time at 0.75×, 1.25×, and 1.75× levels (increases of 190ms, 35ms, and -180ms). There was a larger effect at the extreme 0.5× level (387ms). This may have been influenced by the scaling method used, which kept text size constant (to maintain legibility and pointing time) regardless of scale level. Text labels were thus very close to one another at small scales.

Stretching (Figure 8.7e,f), like *scaling*, showed relatively small absolute time increases for most levels (<250ms for all but 0.5× and 2× horizontal stretching, which exceeded 550ms). The higher time for 0.5× horizontal stretching can be explained in the same way as scaling above; the 2× result is reasonable (and the lower time of the higher 2.5× level may be due to the proximity of the screen edge assisting reorientation).

Rotation had much larger effects on reorientation time (Figure 8.7d), with absolute mean time increases from 434ms (30°) to 2284ms (180°). Some of this time will be incurred by reading rotated text, but it is likely that most of it can be attributed to mental processes of reorienting to the rotated frame of reference. This is supported by prior work from Cooper [37], which showed that the time taken to interpret rotated pattern stimuli increases linearly with rotation angle. Linear regression of the reorientation time data with degree of rotation supports Cooper's finding ($R^2 = 0.98$).

Horizontal *perspective* changes (Figure 8.7g), like *rotation*, resulted in an ap-

proximately linear increase in reorientation time across angle ($R^2 = 0.84$). However, the absolute value of the increase (compared to the baseline) was much smaller than rotation (ranging from 226 to 665ms). Vertical perspective changes (Figure 8.7h) had a much smaller effect on reorientation time, ranging from 2ms at 15° to 139ms at 60° (and a negative effect of 152ms at 30°). One possible explanation is that this type of perspective is common in everyday life (e.g., reading on a flat table); regardless, subjects were quickly able to reorient to the transformation.

Learning

During the learning block, participants selected the target items five times each in a random order from a maximally transformed window. Reorientation time data (selection time minus pointing time) is analysed using a 7×5 ANOVA for factors Transformation Type and Repetition. There was a significant effect of Transformation Type ($F_{6,\,78} = 5.1$, $p < 0.001$), largely due to the slow performance of *rotation* (1593ms) with all other transformations within 921±164ms. There was also a significant effect of Repetition ($F_{4,\,52} = 5.9$, $p < 0.005$), with mean reorientation times quickly improving from 1318ms in the first selection to a minimum of 879ms in the third (within 24ms of the time with untransformed windows in the training block). Participants' performance with stable transformed windows quickly matched that of untransformed views. There was no interaction between Transformation Type and Repetition ($F_{24,\,312} = 1.1$, $p = 0.35$), giving no evidence that any transformation type is harder to learn.

8.2.7 Discussion

To summarise, this experiment analysed how quickly users can reorient their expectation for the location of known targets when spatially consistent displays undergo certain transformations (translation, rotation, scaling, stretching, and perspective). Results showed that users can quickly adapt to all forms of transformation (much more quickly than the time needed to find unknown items in the display). Adapting to rotations was much slower than the other transformations (at 180°, 20x that of translation). Results were also replicated showing that rotation reaction times are a linear function of angle.

These results provide a new human-factors characterisation of performance with common display transformations; in addition, the study provides design insights that are deployed in the next study. In particular, the fast reorientation times in response to scaling and stretching suggest that users will be much faster when a spatially consistent approach is used to deal with transformation, than with approaches that rearrange items to fill the transformed window.

8.3 Experiment 8B: Scaling vs. Reflow

Experiment 8A demonstrated that people are able to quickly select familiar items after a spatially consistent display is transformed. Experiment 8B tests the application of this finding in a realistic interface.

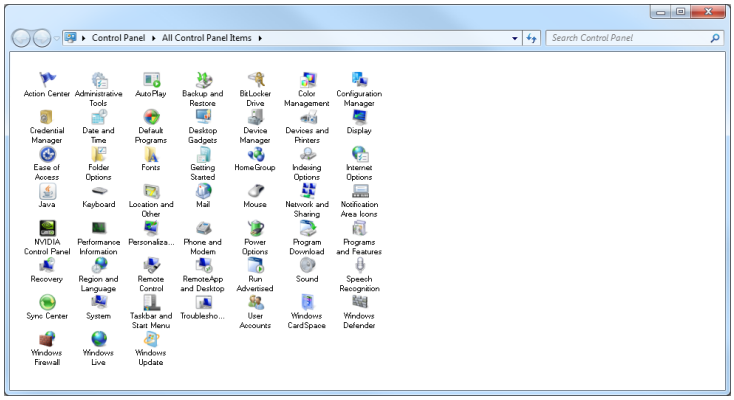
Many commercial interfaces, such as toolbars and file browsers, use a 'reflow' algorithm to rearrange items when the window dimensions change (Figure 8.8c). However, when items are rearranged in this manner, people lose their spatial knowledge of the interface, potentially slowing retrieval. Experiment 8B therefore compared a reflow-based layout strategy to two different spatially consistent designs.

8.3.1 Interface Layout Designs

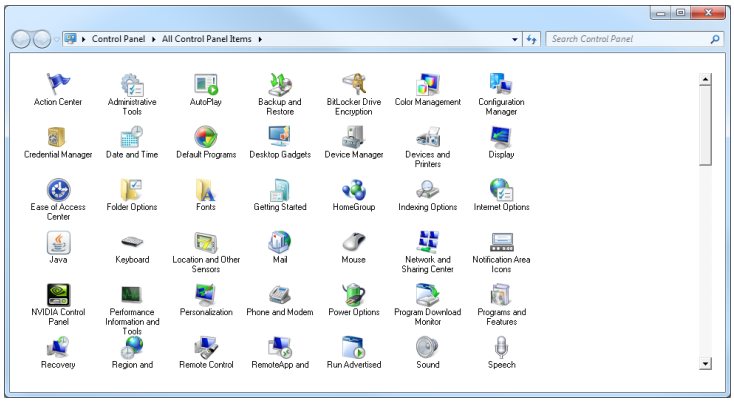
Three designs were considered for adapting the layout of a simple icon view (e.g., a file browser) to window size.

Scaling

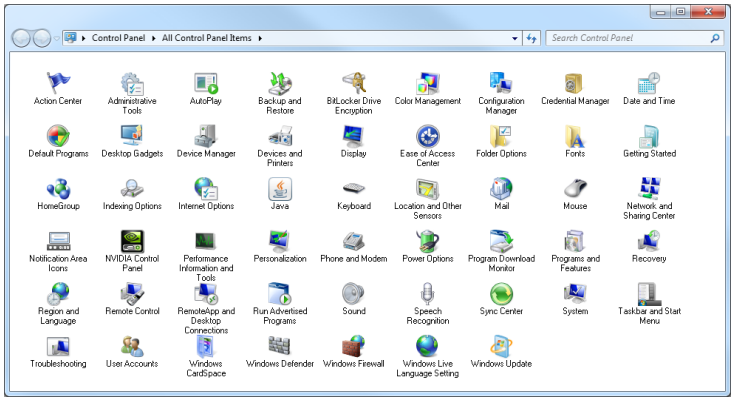
This layout scales a grid of icons to fit the window bounds. Note that when the window's aspect ratio is changed, spatial consistency is maintained relative to the perceived bounds of the item grid, rather than the window edges (Figure 8.8a). An alternative version of this design which stretches the item grid to fill the window bounds (see Figure 8.9) was also considered, but informal testing suggested that users had trouble noticing that the grid was spatially consistent (i.e., they assumed it had been reflowed as normal, and were not as quick to find items).



(a) Scaling



(b) Scrolling



(c) Reflow

Figure 8.8: The three alternative icon layout strategies in a wide window configuration.

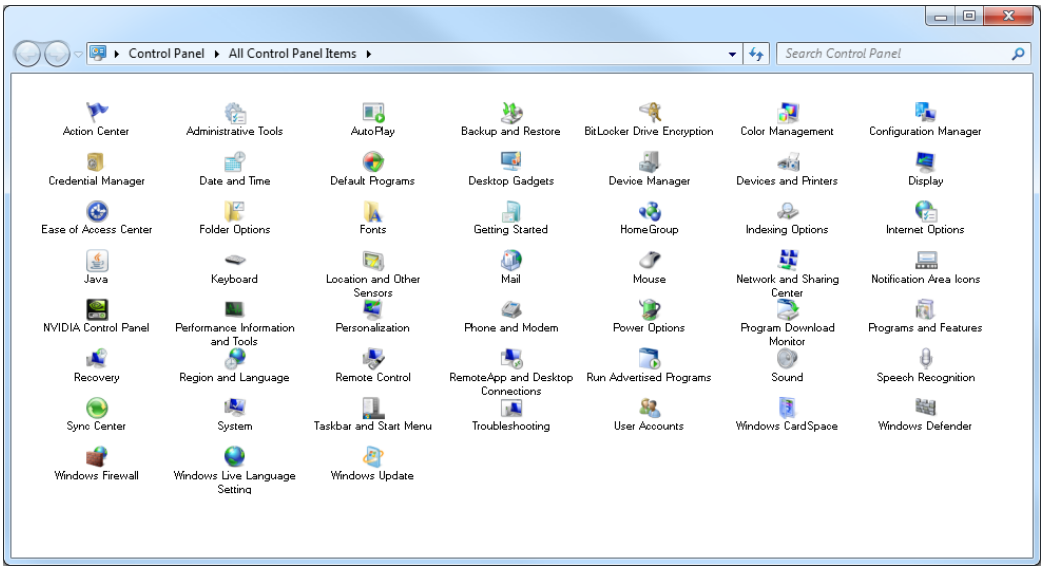


Figure 8.9: An icon layout based on stretching, rather than scaling (not studied).

Scrolling

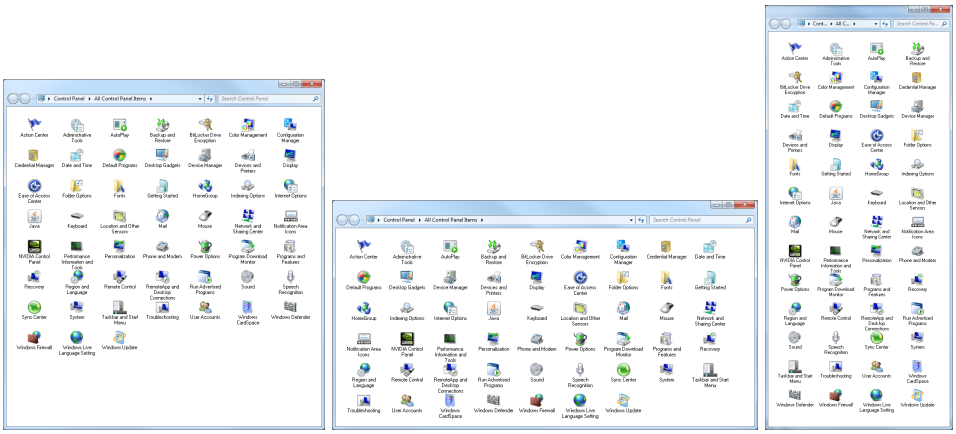
This layout maintains spatial consistency to the original frame of reference, using scrolling to allow viewport translation over the icon grid (Figure 8.8b). The location of items is predictable as an absolute displacement from the information space's origin, but the interface does not maintain relative spatial consistency with respect to the new frame of reference. Scrolling requires more interface manipulations to select targets than the other conditions.

Reflow

This is the standard layout strategy employed in contemporary file browsers: when the window changes size, icons are rearranged to fill the window, in reading order (Figure 8.8c). Reflowing makes efficient use of display space, but requires scrolling when icons do not fit the view.

8.3.2 Procedure

The experimental task consisted of a sequence of selections from a file-browser-like interface, populated with items from the Windows 7 control panel (Figure 8.11). Participants clicked a button to begin each trial, triggering the display of a



(a) Square (b) Wide (c) Tall

Figure 8.10: The three window configurations, using a *reflow* layout strategy.

stimulus in a sidebar. Selecting the target item completed the task and redisplayed the "Click to begin" button.

Two blocks (*training* and *recall*) were completed with each of the three layouts (*scaling*, *scrolling*, and *reflow*). The *training* block consisted of six repetitions of each of six target items, using a *square* window size with a content area of 700×700 pixels. In the *recall* block, the window bounds were varied on every trial to be either *square*, *wide* (917×401px), or *tall* (401×917px). The three configurations are shown in Figure 8.10. The *tall* configuration extended to the vertical height of the monitor, and was just wide enough for all of the items to fit into the reflow window without scrolling. The *wide* configuration was the transposition of *tall*. When window configuration changed, items were arranged according to the layout strategy (*scaling*, *scrolling* or *reflow*). With *square*, the three layout strategies were equivalent. Figure 8.8 shows the effect of each layout strategy on a *wide* configuration. For *scaling*, *wide* scaled the icon grid by 0.7×, and *tall* by 0.57×.

Target items and window configuration sequences were different for each participant in each condition, and targets were selected such that no two target items were in the same row or column. The row and column constraint was used (without subject knowledge) to give an approximately uniform spatial distribution of items in the scrolling condition (to control the number of items that required scrolling).

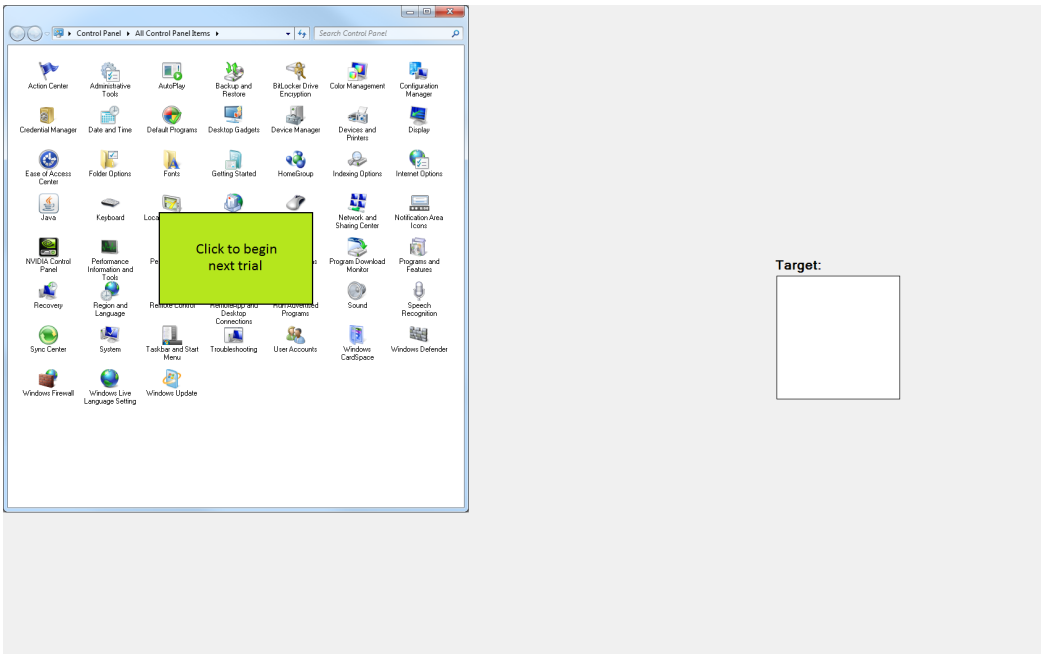


Figure 8.11: The system used in Experiment 8B. Targets were displayed on the right, and participants selected the target items from the interface on the left.

Each participant therefore performed 162 trials:
3 layout strategies $\times$ 2 blocks
training: 36 selections (data discarded)
recall: 18 selections

Participants completed NASA-TLX [80] worksheets and responded to visual appeal questions after each layout. They ranked the layouts for preference, speed and error rate at the end of the experiment.

8.3.3 Participants and Apparatus

15 participants were recruited for the study, with 14 completing it directly after Experiment 8A. One extra participant was recruited to balance the design. Experiments 8A and 8B used the same hardware and setup.

8.3.4 Design and Hypotheses

The experiment was designed as a 3$\times$3 RM-ANOVA for factors Layout (*scaling*, *scrolling*, *reflow*) and Configuration (*square*, *wide*, *tall*), with selection time as the

dependent variable. Layout was counterbalanced using a Latin square.

The primary hypotheses were as follows:

H1: Scaling will be faster than scrolling and reflow. Scaling keeps items spatially consistent, unlike reflow, and requires no extra user action, unlike scrolling, so it is likely to be more efficient.

H2: Scaling will be subjectively preferred by participants. Since scaling disrupt users' spatial memory to a lesser extent than reflow, and requires less mouse input than scrolling, it is likely to be preferred by participants.

8.3.5 Results

Error rates were low in all conditions: 1.5% for *scaling*, 2.6% for *scrolling*, and 1.9% for *reflow*. Trials including incorrect selections were excluded from the analysis; this did not affect the significance of the results. For significant ANOVA effects, partial eta-squared (η^2) is included as a measure of effect size.

Selection Times

Mean selection times were fastest with *scaling* (2.27s, s.d. 0.76), followed by *scrolling* (2.96s, s.d. 1.26) and *reflow* (3.158s, s.d. 1.69), giving a significant main effect of Layout: $F_{2,\,28} = 7.3$, $p = 0.003$, $\eta^2 = 0.34$. With the *scaling* layout, mean selection times following *wide* and *tall* view transformations increased by 262ms and by 277ms over the time taken with the *square* view used for training. These small increases contrast with the substantial increases of 1039ms and 1653ms with the *reflow* layout. Posthoc Bonferroni-adjusted pairwise comparisons ($\alpha = 0.05$) showed that *scaling* was significantly faster than both *scrolling* and *reflow*, but there was no difference between *scrolling* and *reflow*. This result supports **H1**.

As expected, there was a significant effect of Configuration ($F_{2,\,28} = 24.8$, $p < 0.001$, $\eta^2 = 0.64$) with *square* (2.11s, s.d. 0.83) faster than *wide* (2.81s, s.d. 1.05) and *tall* (3.47s, s.d. 1.65). More importantly, there was a Layout×Configuration interaction ($F_{4,\,56} = 5.98$, $p < 0.001$, $\eta^2 = 0.30$), as shown in Figure 8.12: *scaling* performed similarly to *scrolling* and *reflow* in the *square* configuration, but was faster in the *wide* and *tall* configurations.

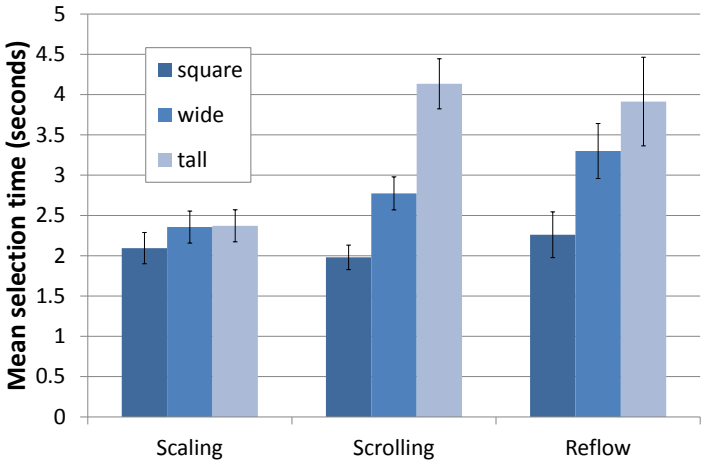


Figure 8.12: Mean selection times in Experiment 8B.

26% of *scrolling* trials in the *wide* configuration required the user to scroll the viewport, with 18% for *tall* and 0% for *square*. Figure 8.12 shows that *scrolling* was slowest in the *tall* condition, which required horizontal scrolling.

Subjective Responses

TLX questionnaire responses showed no significant differences, perhaps due to low statistical power; further study in this area is needed. **H2** is therefore unsupported.

Participant comments revealed some interesting issues with the designs. Four participants mentioned it was more difficult to identify the smaller icons in the scaling condition, making them harder to find. P6 mentioned that the text was hard to read, but that having the icons present helped. One way to alleviate this problem could be to keep font size constant during scaling, to allow for easier recognition of items.

Regarding the reflow condition, participants liked that it maintained the size of icons, but disliked that the icons moved: P1 said ”it creates difficulty to find the item, but not too much”, and P11 said ”the location changes made it hard to get the item”.

Opinions on the scrolling interface were mixed. Two participants said they liked it: P11 said ”the location of items have not changed and the sizes are the

same, so it's better for me to find", and P1 said "it's OK if you're familiar with Word and Excel". In contrast, P12 wrote that it was "very annoying" and P2 wrote that scrolling made it "hard to find things". P5 and P6 both mentioned that scrolling vertically, which could be done with the scroll wheel, was much easier than scrolling horizontally, which required dragging the scroll thumb.

8.4 Discussion

Experiment 8A showed that users can quickly adapt to many forms of view transformation, allowing rapid selections when items remain spatially stable within the frame of reference. Experiment 8B used this finding to compare user performance between the contemporary 'reflow' strategy (which reduces spatial stability) and a scaling layout that maintains spatial stability. Results confirmed that selection times following view transformations were much faster with the scaling layout than with the reflowing layout.

This section discusses the implications of these findings, as well as the limits of their applicability.

8.4.1 Generalising the results of Experiment 8B

Experiment 8A examined human performance factors in response to view transformations, requiring an experimental method that was substantially abstracted away from day to day interaction contexts. Experiment 8B, in contrast, focused on a specific interaction context (contemporary icon layouts in file browsers), but in doing so necessarily introduced potential confounds, including icon visual saliency, dataset ordering and size, and specific transformation settings.

Experiment 8B used the actual icons from the MS Windows control panel, and the reflow condition emulated its behaviour. This was done to assist external validity, while aware of the differing visual salience across icons – for example, the colorful 'Default Programs' icon is more likely to pop out than the small, grey 'keyboard' icon. The experimental design mitigated these effects by randomising the target set for each participant and layout.

The method also used an alphabetical ordering of icons in each view, again to maintain consistency with the current Microsoft layout. This arrangement is very likely to have assisted users in identifying target locations after reflowing.

However, logical or predictable icon arrangements are challenging to determine in many contexts, and it is likely that the benefits of scaling would be more pronounced if predictable ordering was unavailable.

The size of the dataset was also determined by the typical size of the control panel window. The sizes of the tall and wide windows were selected to maximally utilise space in the reflowing condition without scrolling – i.e., the sizes were biased in favour of reflowing. If the windows had been any smaller, the reflowing condition would have required scrolling. There are, however, interesting questions for the scaling condition around the relationship between performance and scale factor. Experiment 8A suggested that performance deteriorates as views are transformed further from $1\times$ size, and there are likely trade-offs between the costs of reducing scale and the costs of increased scrolling. Further work in this area is required.

Finally, subjective responses in Experiment 8B showed no significant differences between conditions. Participants were neither strongly in favour of nor strongly opposed to the scaling view, but it is unclear how their opinions would change if, for example, more extreme scale factors were used, or if the reflowing condition had required scrolling. Again, further work is needed, but it currently appears that scaling allows much faster performance following transformations than does reflowing, and that this benefit comes without the costs of negative subjective reaction.

8.5 Conclusions

Spatial consistency is a powerful organizing principle for interfaces, but everyday use involves many forms of view transformation. A study was conducted to improve understanding of how performance with spatially consistent views is influenced by different forms and magnitudes of display transformation. Results showed that users can quickly reorient their spatial understanding to all of the tested transformations, but that adaptation to rotation is much slower than the others. These findings were then tested in a real-world usage scenario, with the hypothesis that performance with a file browser could be improved by replacing the reflow layout approach with a layout that scaled the view. Results showed substantially improved performance. The primary design implication of this chapter

is that spatial consistency should be used as a fundamental design principle for interfaces and information displays.

Part IV

General Discussion and Conclusions

Chapter 9

Discussion and Directions for Future Work

Human spatial memory is one of the primary mechanisms by which users develop expertise with software interfaces, since memory for object locations allows users to access functionality quickly. Spatial memory also develops rapidly, lasts a long time, and has the ability to store many locations, routes, and layouts at once; in short, it is highly capable. Motivated by this fact, this thesis has investigated new ways to design interfaces based on the strength of human spatial memory.

The work presented in this thesis is intended to inform other researchers and practitioners of the performance benefits derived from explicitly designing for spatial memory. The CommandMap interface (Chapters 5 and 6) relies on the strength of spatial memory to enable rapid command selection for experts. The StencilMap experiments (Chapter 7) investigate an in-place technique of presenting command subsets, allowing users to remember item locations when moving to the full interface. The experiments on spatially consistent transformations (Chapter 8) provide baseline data on the ability of users to maintain spatial memory after different kinds of geometric transformations, as well as validating a new, spatially-consistent technique for responding to changes in window size.

However, there is much more that can be done in terms of studying and designing for spatial memory in HCI. This chapter addresses some of these ideas, including directions for extending and generalising the work presented in this thesis, as well as ideas for adding to the knowledge around spatial memory in general. The sections below describe the progress on the research objectives, discuss and generalise the results of the thesis, and provide opportunities for future work.

9.1 Progress on research objectives

The general goal of this thesis was to understand spatial memory in the context of HCI and use that understanding to design efficient user interfaces. As outlined in Chapter 1, that goal was expanded into three concrete research objectives:

1. Understand the capabilities and characteristics of spatial memory in the context of user interfaces.

2. Exploit the capabilities of spatial memory to design, implement, and evaluate efficient command-selection interfaces.

3. Design and evaluate new interface techniques that maintain spatial memory in situations when it would normally be disrupted.

Chapter 1 also presented criteria for measuring the successful completion of these objectives. Below, these criteria are revisited in light of the work performed and used to measure the progress achieved for each objective.

In order to improve understanding of spatial memory in UIs, Objective 1 required the compilation of a comprehensive literature review. This objective was addressed in three parts by Chapters 2, 3 and 4, with Chapter 2 presenting psychological models and theories of spatial memory, Chapter 3 looking at empirically observable properties of spatial memory, and Chapter 4 summarising the existing research in HCI on the interactions between spatial memory and different types of user interfaces. Additionally, new understanding of spatial memory was gained from the experiments on spatially-consistent transformations in Chapter 8.

Objective 2 covered the creation of new interface techniques that exploited spatial memory. The success of Objective 2 depended on whether the interface techniques created were able to empirically and subjectively outperform existing techniques. Addressing this objective, Chapters 5 and 6 presented the CommandMap, which was shown across five experiments to be faster than menus and Ribbons for expert use, and subjectively preferred in real-world tasks. Additionally, Chapter 7's StencilMap design exploits spatial memory to create an effective way of presenting command subsets.

Objective 3 required the identification of instances in HCI where spatial memory is disrupted, and the development of solutions to improve performance in

such situations. The StencilMap (Chapter 7) attempted to address this problem in subset interfaces, which often disrupt spatial memory by moving controls to different locations, by providing a spatially-consistent overlay to highlight important controls instead. Chapter 8 further addressed this objective by studying the case where window content is moved around in response to changing window geometry, providing a spatially-consistent solution that scaled window contents rather than reflowing them. The scaling solution was shown to outperform reflowing when users were familiar with item locations.

In summary, the work presented in this thesis addresses all three of the stated goals: it adds to the body of knowledge surrounding spatial memory in HCI; it presents two new interfaces that exploit spatial memory to improve efficiency over standard interfaces; and it identifies two situations where spatial memory is disrupted by existing interfaces, leading to the development of more efficient, spatially consistent interface techniques.

9.2 Generalisation of results and future work

The research in this thesis consisted of experiments on several specific systems and techniques. However, in each case, the knowledge gained from these experiments is generalisable beyond the specific systems studied. The sections below discuss the wider applicability of the results obtained for CommandMaps, StencilMaps, and spatially consistent transformations, and identify opportunities for future work.

9.2.1 CommandMaps

The design of the CommandMap was based on two simple principles: first, that removing hierarchy navigation improves mechanical command selection time; and second, that users' spatial memory allows them to rapidly locate controls when the locations of those controls remain spatially constant. The results of the CommandMap experiments – Experiment 5A in particular – lend support to the importance of these ideas, suggesting that they could be applied to other areas of interaction.

However, the experiments presented in Chapter 6 also identified some weaknesses and opportunities for improvement in the CommandMap design. This section discusses these opportunities: overcoming initial negative reactions, alter-

native invocation techniques, and possibilities for integrating hotkey labels. The section then summarises the situations in which CommandMaps are most useful, extracts some key lessons for menu design, and finishes with a discussion of other areas in which the design principles of CommandMaps could be used to improve interaction efficiency.

Overcoming initial reactions

One of the key causes for concern when deploying CommandMaps in the real world is that users may have a negative initial reaction to the new command layout. This concern arose in response to participant comments in Experiment 6B: eight of the twelve participants made some sort of reference to an initial impression of visual overload. Although participants' perceptions quickly changed after use, this data will be of little reassurance to a software vendor if they suspect that customers may purchase a competitor's product due to concerns about the CommandMap when viewing it in a shop or online.

While this is a legitimate concern, it is a concern that is often encountered when a vendor wishes to improve the user's experience through interface revisions [106] (i.e., it is not specific to CommandMaps). Primary tools for overcoming such concerns are advertising and consumer education, although product review websites, blogs, and social media provide new avenues for rapid dissemination of positive (and negative) user experiences. To mitigate the effect somewhat, CommandMaps can also be designed to maximise knowledge transfer from existing interfaces. For example, the Word CommandMap implementation presented in Chapter 6 maintains the item layout within each category group, and the default display of the Home tab (i.e., when the CommandMap is inactive) is identical to the default display of the Ribbon.

CommandMap invocation

In both of the CommandMap systems presented in Chapter 6, the CommandMap was displayed by pressing the `Ctrl` key, which caused minor usability problems for several of the participants – particularly when issuing keyboard shortcuts. Repurposing a less-frequently used key such as `Alt` as the activation key would reduce the incidence of overloaded interface actions. An attractive supplemental

method for activating the CommandMap is to use a dedicated button on a mouse, which would allow for unimanual interaction. Many mouse input devices already support configurable buttons, and repurposing one of these for CommandMap activation would be a natural choice, particularly if CommandMaps were widely deployed.

CommandMaps and hotkeys

There are also interesting possibilities in moving CommandMap item selection onto the keyboard. Recent studies demonstrated the effectiveness of ExposeHK [136], which overlays visual controls with their hotkeys when a modifier key is pressed. CommandMaps would allow all hotkeys for the interface to be browsed at once, possibly with the CommandMap invoked by pressing `Ctrl`, and the hotkey overlay by additionally pressing `Alt`. Command selection could then be completed either by clicking as normal with the mouse, or by pressing the associated hotkey. This should also improve the learnability of hotkeys over the Ribbon, by allowing hotkey browsing without first having to move the cursor over the target.

Lessons for menu design

As discussed in Chapter 4.1.2, several studies of menu selection time with hierarchies of different breadth and depth have shown that selection times follow a U-shape with depth, suggesting that menus should be broad, but not too broad (e.g., Miller [143]). However, the success of CommandMaps suggests that selection times can be reduced by making the hierarchy as broad as possible, reducing hierarchy depth to a single level. The results of Chapters 5 and 6 support Cockburn and Gutwin's [29] assertion that when users know item locations or can predict them based on some structure (e.g., alphabetical ordering or semantic categories), performance is optimized by flat command structures; studies such as Miller's [143]) showed different results because they used random command arrangements and did not allow users enough time to develop item location memory. A more complete analysis of menu layout studies is provided by Cockburn and Gutwin [29].

More generally, these studies support the idea that spatial memory should be prioritised over visual search when optimising control layouts. Current cogni-

tive theory [33] suggests that visual search is the primary location mechanism for novice users, and that retrieval from spatial memory is used when available. However, since spatial memory develops rapidly, users spend more of their lifetime with an application retrieving locations from memory than searching for them.

A final design lesson can be drawn from the fact that participants in Experiment 6B did not consider the modal nature of the CommandMap to be an impediment to use. This implies that with careful design (i.e., using translucency and a quasi-modal activation key), future interfaces can successfully use a modal separation between content and interface controls to allow more screen real estate to be devoted to both.

In what situations are CommandMaps most useful?

CommandMaps provide two primary advantages over menus and ribbons for organizing commands in graphical user interfaces. First, they help novice or intermediate users gain an overview of an application's functionality. Second, they increase the efficiency of expert users who have learned item locations. In general, CommandMaps are best in situations when all, or most, of an application's commands can fit on a single screen, and when the command set would otherwise be large enough to necessitate partitioning into multiple menus and ribbons.

In applications with extremely large command sets, however (such as Autodesk Maya or 3ds Max), providing a CommandMap containing all of the functionality of an application may be infeasible. In these circumstances, there are several ways a CommandMap could still be useful. First, it could provide a broad top-level hierarchy to the command set, similar to Maya's Hotbox [116]. Second, it could be provided as a shortcut interface, allowing the user to customise the CommandMap with the tools they most frequently use. Alternatively, it could contain a subset of the controls most frequently needed by novice users, providing access to a basic overview of the application's functionality.

Other applications

The CommandMap design principles of flattened hierarchies and spatially constant layouts are also applicable in other areas. As discussed in Section 4.1, they have already been used to improve window switching [191, 192], within-

document navigation [32], and scrolling lists [74]. However, there are still more opportunities to make use of these principles. One promising avenue for research is mobile interaction. Gutwin et al.'s FastTap system (Appendix B) is one example of a flat, spatially-stable interface for mobile, although at the time of writing, there are still many possible improvements to be made.

While work thus far has focused on everyday desktop and mobile interaction, there are also many domain-specific areas that have not yet been investigated. High-impact applications where efficiency is important, such as financial software, could perhaps benefit from CommandMap-like design. Interfaces for public spaces, such as kiosk software, could also potentially be improved by CommandMaps, although success in this area most likely depends on the frequency with which individual users interact with the system (i.e., systems that are only used once or twice by individual users may benefit less from the addition of CommandMaps, since the users will be unable to develop a memory for item locations).

9.2.2 StencilMaps

The StencilMap design, presented in Chapter 7, was intended to help novice users to find commands they were likely to need, while at the same time allowing them to build up a knowledge of item locations. The StencilMap was partially successful in these two goals: it was shown to be better than ephemeral highlighting and a basic CommandMap for novice visual search (though worse thana simple command palette), and better than the palette for longer-term location learning (though worse than ephemeral and the basic CommandMap).

These results suggest that there is a fundamental trade-off between initial ease of use and spatial learning, and that the StencilMap represents a compromise between these two goals. This section discusses this trade-off and its applicability in general, and identifies some potential future research directions for StencilMaps.

The tradeoff between ease of use and learning

The StencilMap experiments, particularly Experiment 7A, suggested that making things easier for new users has the side-effect of reducing their spatial learning. This result is consistent with prior experiments by Ehret et al. [48] and Cockburn et al. [34], both of which showed that decreased visual feedback increases users'

reliance on their spatial memory, thus improving their learning.

The fact that this trade-off exists raises the question: how much assistance should an interface provide (or, conversely, how effortful should the interface be)? The answer is likely to vary with the goals of the system and its users. For training games like Cockburn's [34], it makes sense to provide a more effortful interface, since spatial learning is the explicit goal of the application. But for regular command interfaces, where the goal is productivity, the answer is less clear. One way to determine whether or not an assistive system (such as StencilMaps) is worth using would be to empirically compare the overall productivity of users who were given the assistance to begin with against users who were not. Users who were given the assistance (assuming it was effective) would presumably experience an initial efficiency boost, but be less efficient over the long term; while users who were not given the assistance would presumably experience the opposite. However, determining which approach saves more time overall in a work environment would likely be a costly and time-consuming exercise.

Another factor to take into account is whether or not the assistance is permanent, or whether it only exists temporarily to aid new users. If the assistance is permanent, then long-term learning is less of a concern since users can always make use of the assistive interface to be efficient. For example, a permanent palette of commands is likely to be more effective than a permanent StencilMap. However, it is worth noting that there is a more subtle concern with palettes that copy commands into more than one location – the increased user choice creates *decision costs* [164] that may slow down users (i.e., extra time is required for the user to decide which of the two options will be most efficient to use).

Future directions for StencilMaps

There are several design options for StencilMaps that have not yet been evaluated. For example, the strength of the StencilMap's "pop-out" effect may be increased by combining it with the ephemeral highlighting technique (i.e., initially showing a completely black stencil that gradually fades out to 30% opacity). This would keep the 'permanence' advantage of the original StencilMap design, while potentially increasing the visual pop-out effect for items in the highlighted set.

Another potential improvement could be to automatically vary the amount of

time for which StencilMaps are visible. For example, the system could automatically detect when the user has sufficient knowledge of the highlighted items, and automatically remove the highlighting. However, the user would have to be educated about this feature so it does not occur unexpectedly.

Finally, although primarily evaluated in this thesis in the context of training, StencilMaps could also be combined with a predictive algorithm (such as AccessRank [58]) to create an adaptive interface that highlights commands the user is most likely to need. Further research would be needed to determine the effectiveness of this approach compared to other adaptive systems.

9.2.3 Spatial consistency

As evidenced by the existing literature presented in Chapter 4, keeping items in spatially constant locations is extremely important for the development of spatial memory. Chapter 8 further investigated this idea and showed that if spatial consistency can be maintained *relative to a frame of reference*, several types of geometric transformations (translation, scaling, stretching, and perspective change) can be performed without substantially disrupting the user's spatial memory.

As discussed in Chapter 8, this result allows for the design of new solutions to situations in which spatial memory is normally disrupted, such as window resizing. However, while these solutions may be better for spatial memory, they often do so at the expense of other design guidelines. The sections below address this issue, as well as discussing the limitations of scaling-based designs and suggesting other areas in which the principle of spatial consistency could be used to improve design.

Conflicts and trade-offs with other design guidelines

Following the guideline of spatial consistency allows for the preservation and development of spatial memory. However, there are some situations where the guideline conflicts with others. For example, the file browser design presented in Chapter 8 (and shown again in Figure 9.1) maintains spatial consistency, but adds a section of empty space to the window. Similarly, applying spatial consistency to rotation on the iPad may result in squashed-looking layouts that are less visually pleasing. In cases like this, designers should weigh up the relative merits

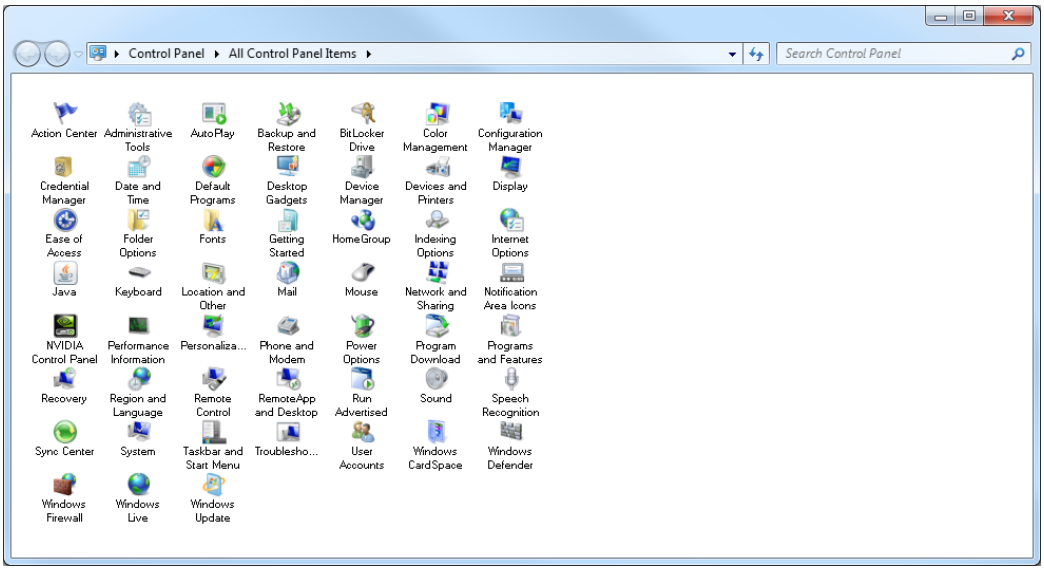


Figure 9.1: The scaling file browser solution from Chapter 8 maintains spatial consistency at the cost of aesthetics.

of maintaining spatial consistency. If the application is designed for productivity rather than recreation, for example, spatial consistency should be given more consideration than purely aesthetic concerns.

Another factor worth considering is how often transformations are likely to take place. In windows that are infrequently resized, such as dialog boxes, spatial consistency may not be worth considering at all. In situations like the iPad rotation, where only two distinct layouts are possible, designers may choose to simply let users learn two different layouts rather than attempting to maintain spatial consistency between them. In general, more work is needed to determine the relative importance of different design guidelines and their effect on productivity and user satisfaction.

Limitations of scaling-based designs

In some situations, the scaling grid solution presented in Chapter 8 may be impractical. For example, when there are many items in an interface, or when the window is small, scaling the entire icon set to fit the window bounds will result in text and icons that are too small to read. Furthermore, pointing can be difficult at very small scales. In these scenarios the best solution is likely to be a hybrid

scaling/scrolling strategy, where the grid is scaled according to the width of the window and a vertical scrollbar allows users to access off-screen items. When the window width becomes too small to feasibly scale items, scaling ends and a horizontal scrollbar can be added. It is worth noting that as sets grow, the problems of 'reflowing' also increase (i.e., items near the end of the list will be even further displaced from their original locations). Further work in this area is needed.

There are other situations where scaling alone does not provide complete spatial consistency. In particular, Experiment 8B studied an icon set that changes slowly if at all (control panel icons); in windows where content changes more quickly, with frequent additions and deletions, maintaining spatial consistency becomes more complex. In these situations, there are several possible solutions. First, if the content changes slowly enough, it is likely that the user's spatial memory will be able to keep up with the changes [191], so no special changes would be needed. Second, users could be in charge of placing new items in a display (as with a phone's home screen, or in systems like the Data Mountain [169]); in these cases, the act of placing the items can help to overcome the difficulties caused by changing content. Third, ordering by addition (i.e., new icons are added at the end of the display) would lead to stable arrangements that allow the development of spatial memory. Fourth, a spatial organization could be used as one of several view options: in situations where content changes slowly, the user would gain the benefits of developing spatial memory; in situations where items change frequently, the user could switch to an alphabetic arrangement (or a list view).

Other applications of spatially consistent design

The primary design implication of Chapter 8 was that spatial consistency should be a fundamental consideration in the design of interfaces and information displays. In many cases, altering existing interfaces to maintain spatial consistency is a relatively simple matter – for example, on mobile devices that allow landscape and portrait view modes (switched by accelerometer input) interface design could favor relative spatial consistency of items, rather than seeking ways to rearrange interface components to exploit the variable display space in the different layouts.

Designing for spatial consistency allows new styles of interaction, such as those demonstrated by the Data Mountain [169], Space-Filling Thumbnails [32]

or CommandMaps. This type of spatial interface design can also be applied to other domains, such as command selection on mobile devices (Appendix B).

Another interesting possibility lies in creating new interface toolkits and APIs that are more robust to variable display requirements. Built-in scaling functions to accommodate different window sizes and/or display resolutions would greatly facilitate the implementation of interfaces that are spatially robust, rather than resorting to the current methods of reflowing, rearranging, and elision.

9.2.4 Other knowledge gaps in the HCI spatial memory literature

The literature presented in Chapters 2, 3, and 4 reveals many opportunities for further work on spatial memory in HCI. The sections below identify some of those questions.

Decay and interference

Section 2.3.1 discussed different types of forgetting (such as decay over time and interference from other memories) and their impact on spatial memory. However, there is very little research on forgetting in the context of user interfaces. It is therefore natural to ask: how long does spatial memory for a user interface last? Is it as resilient to decay as other spatial memories from everyday life? Furthermore, how does interference affect spatial memory in user interfaces? Can learning the layout of one spatial interface compromise retrieval for another previously-learned interface? Are there factors (e.g., similarity of locations or of icons) that affect this interference? Does allowing users to manually place items benefit spatial learning compared to normal interaction?

Accuracy

Section 3.2.1 discussed systems and studies that measured users' abilities to remember the locations of items they'd never seen. However, such studies are rare, and often confined to the context of specific systems. More general knowledge is lacking: for example, exactly how accurate is egocentric spatial memory? What is the interaction between retrieved approximate memory in visual search? How does this interaction change as users progress toward expertise? Can spatial memory be improved by manipulating the positions of boundaries and partitions?

Variation in ability

As discussed in Section 3.5, spatial ability can vary between groups of users; for example, deterioration in ability can occur with age. More research is required to determine whether spatial interfaces (such as those presented in this thesis) are disproportionately more difficult to use for users with low spatial ability, and if so, what techniques could be used to counteract this effect.

Chapter 10

Conclusions

Spatial memory plays an important part in determining whether, and to what degree, users are able to transition to expert behaviour in user interfaces. This thesis has identified two areas where spatial memory is either underutilised or disrupted. First, the CommandMap was presented as a way of exploiting spatial memory to enable rapid command selection, an area where spatial memory has previously been underutilised. Next, this thesis presented and analysed the StencilMap, a subset interface designed to aid novice users with visual search while supporting the development of location knowledge. Finally, the thesis examined situations where spatial interfaces are re-arranged due to changes in window geometry, and presented a design principle of *spatial consistency* to avoid disrupting spatial memory in such situations.

This thesis has made six primary contributions to knowledge in the domain of spatial memory and command selection interfaces. These are:

1. A review of existing literature on spatial memory as it relates to human-computer interaction. The review covers underlying psychological models of spatial memory, observable properties of spatial memory, and the ways in which spatial memory affects different types of commercial and research UIs. Prior to this work, no such review of spatial memory and HCI existed; the goal of the review is to serve as a guide for future researchers to the current state of spatial memory research, as well as provide guidelines for designers and HCI practitioners.

2. The design, theoretical modelling and evaluation of the CommandMap, an interface that exploits the strength of spatial memory by displaying as many items as possible in a spatially consistent arrangement. The CommandMap

was predicted and shown to be faster and subjectively preferred over cascading menus and the Microsoft Ribbon for expert users, and no different to menus and the Ribbon for novice users.

3. The development of two real-world CommandMap implementations for Microsoft Word and Pinta. These implementations were used to evaluate CommandMaps in more realistic tasks, demonstrating that they maintain their advantages over standard interfaces. In addition, these implementations can be used by other researchers to continue the development of CommandMaps in the future.

4. The design and evaluation of the StencilMap, an extension of the CommandMap that uses a stencil overlay to highlight a subset of relevant commands. The StencilMap was shown to provide faster visual search than standard CommandMaps and ephemeral highlighting, and better spatial learning of the full interface than subsets presented in command palettes. Additionally, the StencilMap evaluations revealed information on the tradeoff between making interaction easier for novices and supporting long-term spatial learning.

5. A human-factors experiment demonstrating the effects of spatially consistent geometric transformations on spatial memory. Translation, scaling, stretching, and perspective changes were shown to have minimal effects on spatial recall times.

6. An experiment demonstrating that a spatially consistent file browser interface allows for faster item selection than one which reflows items when the window size is changed.

Together, this research provides a foundation for further work, both in spatial memory and in the design of efficient interfaces for command selection. The interface designs and experimental results presented in this thesis may be used to inform both designers and HCI researchers as they push the boundaries of efficient user interaction.

Bibliography

[1] Ahlström, D, Cockburn, A, Gutwin, C, and Irani, P. Why it's quick to be square: modelling new and existing hierarchical menu designs. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '10, Atlanta, Georgia, USA, 2010. ACM, pages 1371–1380.

[2] Alan, D, Janet, F, Gregory, A, and Russell, B. *Human–Computer Interaction*. Prentice Hall International, 1993.

[3] Alexander, J, Cockburn, A, Fitchett, S, Gutwin, C, and Greenberg, S. Revisiting read wear: analysis, design, and evaluation of a footprints scrollbar. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '09, Boston, MA, USA, 2009. ACM, pages 1665–1674.

[4] Allen, G. L, Siegel, A. W, and Rosinski, R. R. The role of perceptual context in structuring spatial knowledge. *Journal of Experimental Psychology: Human Learning and Memory*, 4(6):617, 1978.

[5] Anderson, F and Bischof, W. F. Learning and performance with gesture guides. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '13, Paris, France, 2013. ACM, pages 1109–1118.

[6] Andrade, J and Meudell, P. Short report: Is spatial information encoded automatically in memory? *The Quarterly Journal of Experimental Psychology Section A*, 46(2):365–375, 1993.

[7] Baddeley, A. The episodic buffer: a new component of working memory? *Trends in cognitive sciences*, 4(11):417–423, 2000.

[8] Baddeley, A. Working memory: Looking back and looking forward. *Nature Reviews Neuroscience*, 4(10):829–839, 2003.

[9] Baddeley, A. D. *Essentials of human memory.* Psychology Press/Taylor & Francis (UK), 1999.

[10] Baddeley, A. D. The psychology of memory. *The Handbook of Memory Disorders*, page 1, 2003.

[11] Baddeley, A. D and Hitch, G. Working memory. volume 8 of *Psychology of Learning and Motivation*, pages 47 – 89. Academic Press, 1974.

[12] Baudisch, P and Rosenholtz, R. Halo: a technique for visualizing off-screen objects. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '03, Ft. Lauderdale, Florida, USA, 2003. ACM, pages 481–488.

[13] Bederson, B and Boltman, A. Does animation help users build mental maps of spatial information? In *Information Visualization, 1999. (Info Vis '99) Proceedings. 1999 IEEE Symposium on*, 1999, pages 28–35.

[14] Bederson, B. B. The promise of zoomable user interfaces. *Behaviour & Information Technology*, 30(6):853–866, 2011.

[15] Boivin, M. J. The effect of culture on a visual-spatial memory task. *The Journal of General Psychology*, 118(4):327–334, 1991. PMID: 1813596.

[16] Bower, G. H. Analysis of a mnemonic device: Modern psychology uncovers the powerful components of an ancient system for improving memory. *American Scientist*, 58(5):pp. 496–510, 1970.

[17] Brooks, L. R. Spatial and verbal components of the act of recall. *Canadian Journal of Psychology/Revue canadienne de psychologie*, 22(5):349, 1968.

[18] Burigat, S, Chittaro, L, and Parlato, E. Map, diagram, and web page navigation on mobile devices: the effectiveness of zoomable user interfaces with overviews. In *Proceedings of the 10th international conference on Human computer interaction with mobile devices and services*, MobileHCI '08, Amsterdam, The Netherlands, 2008. ACM, pages 147–156.

[19] Byrne, M. D, Anderson, J. R, Douglass, S, and Matessa, M. Eye tracking the visual search of click-down menus. In *Proceedings of the SIGCHI conference on Human Factors in Computing Systems*, CHI '99, Pittsburgh, Pennsylvania, USA, 1999. ACM, pages 402–409.

[20] Card, S. K. Visual search of computer command menus. *Attention and performance X: Control of language processes*, pages 97–108, 1984.

[21] Card, S. K, Moran, T. P, and Newell, A. *The psychology of human computer interaction*. Routledge, 1983.

[22] Carpendale, M, Cowperthwaite, D, and Fracchia, F. Making distortions comprehensible. In *Visual Languages, 1997. Proceedings. 1997 IEEE Symposium on*, 1997, pages 36–45.

[23] Carroll, J. M and Carrithers, C. Training wheels in a user interface. *Communications of the ACM*, 27(8):800–806, August 1984.

[24] Carroll, J. M and Rosson, M. B. *Paradox of the active user.* The MIT Press, 1987.

[25] Chapuis, O and Roussel, N. Metisse is not a 3d desktop! In *Proceedings of the 18th Annual ACM Symposium on User Interface Software and Technology*, UIST '05, Seattle, WA, USA, 2005. ACM, pages 13–22.

[26] Chiasson, S, Biddle, R, and van Oorschot, P. C. A second look at the usability of click-based graphical passwords. In *Proceedings of the 3rd symposium on Usable privacy and security*, SOUPS '07, Pittsburgh, Pennsylvania, 2007. ACM, pages 1–12.

[27] Chrastil, E. R and Warren, W. H. Active and passive contributions to spatial learning. *Psychonomic Bulletin & Review*, 19(1):1–23, 2012.

[28] Cockburn, A, Quinn, P, Gutwin, C, Ramos, G, and Looser, J. Air pointing: Design and evaluation of spatial target acquisition with and without visual feedback. *International Journal of Human-Computer Studies*, 69(6):401 – 414, 2011.

[29] Cockburn, A and Gutwin, C. A predictive model of human performance with scrolling and hierarchical lists. *Human-Computer Interaction*, 24(3): 273–314, 2009.

[30] Cockburn, A and McKenzie, B. Evaluating the effectiveness of spatial memory in 2d and 3d physical and virtual environments. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '02, Minneapolis, Minnesota, USA, 2002. ACM, pages 203–210.

[31] Cockburn, A and McKenzie, B. Evaluating spatial memory in two and three dimensions. *International Journal of Human-Computer Studies*, 61 (3):359–373, 2004.

[32] Cockburn, A, Gutwin, C, and Alexander, J. Faster document navigation with space-filling thumbnails. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '06, Montréal, Québec, Canada, 2006. ACM, pages 1–10.

[33] Cockburn, A, Gutwin, C, and Greenberg, S. A predictive model of menu performance. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '07, San Jose, California, USA, 2007. ACM, pages 627–636.

[34] Cockburn, A, Kristensson, P. O, Alexander, J, and Zhai, S. Hard lessons: effort-inducing interfaces benefit spatial learning. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '07, San Jose, California, USA, 2007. ACM, pages 1571–1580.

[35] Cockburn, A, Karlson, A, and Bederson, B. B. A review of overview+detail, zooming, and focus+context interfaces. *ACM Computing Surveys*, 41(1):2:1–2:31, January 2009.

[36] Cohen, J. *Statistical Power Analysis for the Behavioral Sciences*. Lawrence Erlbaum Associates, Hillsdale, NJ, 2nd edition, 1988.

[37] Cooper, A. Mental rotation of random two-dimensional shapes. *Cognitive Psychology*, pages 20–43, 1975.

[38] Craik, F. I and Lockhart, R. S. Levels of processing: A framework for memory research. *Journal of verbal learning and verbal behavior*, 11(6): 671–684, 1972.

[39] Czerwinski, M, Van Dantzich, M, Robertson, G, and Hoffman, H. The contribution of thumbnail image, mouse-over text and spatial location memory to web page retrieval in 3d. In *Proceedings of INTERACT*, volume 99, 1999, pages 163–170.

[40] Darken, R. P and Sibert, J. L. Wayfinding strategies and behaviors in large virtual worlds. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '96, Vancouver, British Columbia, Canada, 1996. ACM, pages 142–149.

[41] Das, A and Stuerzlinger, W. Unified modeling of proactive interference and memorization effort: A new mathematical perspective within ACT-R theory. In *CogSci 2013*, Berlin, Germany, 2013.

[42] Davis, E. T, Scott, K, Pair, J, Hodges, L. F, and Oliverio, J. Can audio enhance visual perception and performance in a virtual environment? In *Proceedings Of The Human Factors And Ergonomics Society 43rd Annual Meeting*, 1999, pages 1197–1201.

[43] Dixon, M and Fogarty, J. Prefab: Implementing advanced behaviors using pixel-based reverse engineering of interface structure. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '10, Atlanta, Georgia, USA, 2010. ACM, pages 1525–1534.

[44] Doeller, C. F and Burgess, N. Distinct error-correcting and incidental learning of location relative to landmarks and boundaries. *Proceedings of the National Academy of Sciences*, 105(15):5909–5914, 2008.

[45] Duh, H. B.-L, Tan, G. C. B, and Chen, V. H.-h. Usability evaluation for mobile device: a comparison of laboratory and field tests. In *Proceedings of the 8th conference on Human-computer interaction with mobile devices and services*, MobileHCI '06, Helsinki, Finland, 2006. ACM, pages 181–186.

[46] Eals, M and Silverman, I. The hunter-gatherer theory of spatial sex differences: Proximate factors mediating the female advantage in recall of object arrays. *Ethology and Sociobiology*, 15(2):95 – 105, 1994.

[47] Ebbinghaus, H. *Memory: A contribution to experimental psychology*. Number 3. Teachers college, Columbia university, 1913.

[48] Ehret, B. D. Learning where to look: location learning in graphical user interfaces. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '02, Minneapolis, Minnesota, USA, 2002. ACM, pages 211–218.

[49] Elliott, D and Madalena, J. The influence of premovement visual information on manual aiming. *The Quarterly Journal of Experimental Psychology Section A*, 39(3):541–559, 1987.

[50] Elmes, D. G. Interference in spatial memory. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, 14(4):668, 1988.

[51] Engel, F. Visual conspicuity, visual search and fixation tendencies of the eye. *Vision Research*, 17(1):95–108, 1977.

[52] Evans, A and Wobbrock, J. Taming wild behavior: The input observer for obtaining text entry and mouse pointing measures from everyday computer use. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '12, Austin, Texas, USA, 2012. ACM, pages 1947–1956.

[53] Findlater, L and Gajos, K. Z. Design space and evaluation challenges of adaptive graphical user interfaces. *AI Magazine*, 30(4):68–73, 2009.

[54] Findlater, L and McGrenere, J. A comparison of static, adaptive, and adaptable menus. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '04, Vienna, Austria, 2004. ACM, pages 89–96.

[55] Findlater, L and McGrenere, J. Evaluating reduced-functionality interfaces according to feature findability and awareness. In Baranauskas, C, Palanque, P, Abascal, J, and Barbosa, S, editors, *Human-Computer Interaction INTERACT 2007*, volume 4662 of *Lecture Notes in Computer Science*, pages 592–605. Springer Berlin Heidelberg, 2007.

[56] Findlater, L and McGrenere, J. Impact of screen size on performance, awareness, and user satisfaction with adaptive graphical user interfaces. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '08, Florence, Italy, 2008. ACM, pages 1247–1256.

[57] Findlater, L, Moffatt, K, McGrenere, J, and Dawson, J. Ephemeral adaptation: the use of gradual onset to improve menu selection performance. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '09, Boston, MA, USA, 2009. ACM, pages 1655–1664.

[58] Fitchett, S and Cockburn, A. AccessRank: predicting what users will do next. In *Proceedings of the 2012 ACM annual conference on Human Factors in Computing Systems*, CHI '12, Austin, Texas, USA, 2012. ACM, pages 2239–2242.

[59] Fitchett, S, Cockburn, A, and Gutwin, C. Improving navigation-based file retrieval. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '13, Paris, France, 2013. ACM.

[60] Fitts, P. M and Posner, M. I. *Human performance.* Brooks/Cole, 1967.

[61] Fitzmaurice, G and Buxton, W. The Chameleon: spatially aware palmtop computers. In *Conference Companion on Human Factors in Computing Systems*, CHI '94, Boston, Massachusetts, USA, 1994. ACM, pages 451–452.

[62] Fitzmaurice, G. W. Situated information spaces and spatially aware palmtop computers. *Communications of the ACM*, 36(7):39–49, July 1993.

[63] Fitzmaurice, G. W, Zhai, S, and Chignell, M. H. Virtual reality for palmtop computers. *ACM Transactions on Information Systems*, 11(3):197–218, July 1993.

[64] Flaherty, M. The validity of tests of visuo-spatial skills in cross-cultural studies. *The Irish Journal of Psychology*, 18(4):404–412, 1997.

[65] Furnas, G. W. Generalized fisheye views. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '86, Boston, Massachusetts, USA, 1986. ACM, pages 16–23.

[66] Gagnon, D. Videogames and spatial skills: An exploratory study. *Educational Communication and Technology Journal*, 33(4):263–275, 1985.

[67] Gajos, K. Z, Czerwinski, M, Tan, D. S, and Weld, D. S. Exploring the design space for adaptive graphical user interfaces. In *Proceedings of the working conference on Advanced visual interfaces*, AVI '06, Venezia, Italy, 2006. ACM, pages 201–208.

[68] Gajos, K. Z, Everitt, K, Tan, D. S, Czerwinski, M, and Weld, D. S. Predictability and accuracy in adaptive user interfaces. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '08, Florence, Italy, 2008. ACM, pages 1271–1274.

[69] Gibson, B. S and Kelsey, E. M. Stimulus-driven attentional capture is contingent on attentional set for displaywide visual features. *Journal of Experimental Psychology: Human Perception and Performance*, 24(3):699, 1998.

[70] Gray, W. D and Fu, W.-T. Soft constraints in interactive behavior: The case of ignoring perfect knowledge in-the-world for imperfect knowledge in-the-head. *Cognitive Science*, 28(3):359–382, 2004.

[71] Gustafson, S, Baudisch, P, Gutwin, C, and Irani, P. Wedge: clutter-free visualization of off-screen locations. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '08, Florence, Italy, 2008. ACM, pages 787–796.

[72] Gustafson, S, Bierwirth, D, and Baudisch, P. Imaginary interfaces: spatial interaction with empty hands and without visual feedback. In *Proceedings of the 23nd annual ACM symposium on User interface software and technology*, UIST '10, New York, New York, USA, 2010. ACM, pages 3–12.

[73] Gustafson, S, Holz, C, and Baudisch, P. Imaginary phone: learning imaginary interfaces by transferring spatial memory from a familiar device. In *Proceedings of the 24th annual ACM symposium on User interface software and technology*, UIST '11, Santa Barbara, California, USA, 2011. ACM, pages 283–292.

[74] Gutwin, C and Cockburn, A. Improving list revisitation with listmaps. In *Proceedings of the working conference on Advanced visual interfaces*, AVI '06, Venezia, Italy, 2006. ACM, pages 396–403.

[75] H. Logie, R and Pearson, D. G. The inner eye and the inner scribe of visuo-spatial working memory: Evidence from developmental fractionation. *European Journal of Cognitive Psychology*, 9(3):241–257, 1997.

[76] Habib, M and Sirigu, A. Pure topographical disorientation: A definition and anatomical basis. *Cortex*, 23(1):73 – 85, 1987.

[77] Hall, J and Kimura, D. Sexual orientation and performance on sexually dimorphic motor tasks. *Archives of Sexual Behavior*, 24(4):395–407, 1995.

[78] Halpern, D. F. *Sex differences in cognitive abilities*. Lawrence Erlbaum, 2000.

[79] Hansen, W. J. User engineering principles for interactive systems. In *Proceedings of the November 16-18, 1971, fall joint computer conference*, AFIPS '71 (Fall), Las Vegas, Nevada, 1971. ACM, pages 523–532.

[80] Hart, S. G and Staveland, L. E. Development of NASA-TLX (Task Load Index): Results of empirical and theoretical research. *Human mental workload*, 1(3):139–183, 1988.

[81] Hasher, L and Zacks, R. T. Automatic and effortful processes in memory. *Journal of Experimental psychology: general*, 108(3):356, 1979.

[82] Hazen, N. L. Spatial exploration and spatial knowledge: Individual and developmental differences in very young children. *Child Development*, 53 (3):pp. 826–833, 1982.

[83] Hegarty, M, Montello, D. R, Richardson, A. E, Ishikawa, T, and Lovelace, K. Spatial abilities at different scales: Individual differences in aptitude-test performance and spatial-layout learning. *Intelligence*, 34(2):151 – 176, 2006.

[84] Henderson, J. M, Brockmole, J. R, Castelhano, M. S, Mack, M, Fischer, M, Murray, W, and Hill, R. Visual saliency does not account for eye movements during visual search in real-world scenes. *Eye movements: A window on mind and brain*, pages 537–562, 2007.

[85] Hertzum, M. User testing in industry: A case study of laboratory, workshop, and field tests. In *In A. Kobsa & C. Stephanidis (Eds.), User interfaces for all. Proceedings. 5. ERCIM workshop*, 1999, pages 59–72.

[86] Hick, W. E. On the rate of gain of information. *Quarterly Journal of Experimental Psychology*, 4(1):11–26, 1952.

[87] Hill, W. C, Hollan, J. D, Wroblewski, D, and McCandless, T. Edit wear and read wear. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '92, Monterey, California, USA, 1992. ACM, pages 3–9.

[88] Hinckley, K, Pausch, R, and Proffitt, D. Attention and visual feedback: the bimanual frame of reference. In *Proceedings of the 1997 symposium on*

Interactive 3D graphics, I3D '97, Providence, Rhode Island, USA, 1997. ACM, pages 121–ff.

[89] Hochheiser, H and Shneiderman, B. Performance benefits of simultaneous over sequential menus as task complexity increases. *International Journal of Human-Computer Interaction*, 12(2):173–192, 2000.

[90] Hole, G. Decay and interference effects in visuospatial short-term memory. *Perception*, 25(1):53, 1996.

[91] Hornbæ k, K, Bederson, B. B, and Plaisant, C. Navigation patterns and usability of zoomable user interfaces with and without an overview. *ACM Transactions on Computer-Human Interaction*, 9(4):362–389, December 2002.

[92] Hornof, A. J. Cognitive strategies for the visual search of hierarchical computer displays. *Human-Computer Interaction*, 19(3):183–223, 2004.

[93] Hornof, A. J and Kieras, D. E. Cognitive modeling reveals menu search in both random and systematic. In *Proceedings of the ACM SIGCHI Conference on Human factors in computing systems*, CHI '97, Atlanta, Georgia, USA, 1997. ACM, pages 107–114.

[94] Howes, A, Payne, S. J, and Woodward, A. The trouble with shortcuts. In *CHI '00 Extended Abstracts on Human Factors in Computing Systems*, CHI EA '00, The Hague, The Netherlands, 2000. ACM, pages 267–268.

[95] Huttenlocher, J, Hedges, L. V, and Duncan, S. Categories and particulars: Prototype effects in estimating spatial location. *Psychological Review*, 98 (3):352 – 376, 1991.

[96] Hyman, R. Stimulus information as a determinant of reaction time. *Journal of experimental psychology*, 45(3):188, 1953.

[97] Igel, A and Harvey, L. O. Spatial distortions in visual perception. *Gestalt theory*, 13(4):210–231, 1991.

[98] Ion, A, Chang, Y.-L. B, Haller, M, Hancock, M, and Scott, S. D. Canyon: providing location awareness of multiple moving objects in a detail view on large displays. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '13, Paris, France, 2013. ACM, pages 3149–3158.

[99] Ishikawa, T, Fujiwara, H, Imai, O, and Okabe, A. Wayfinding with a gps-based mobile navigation system: A comparison with maps and direct experience. *Journal of Environmental Psychology*, 28(1):74 – 82, 2008.

[100] Itti, L and Koch, C. A saliency-based search mechanism for overt and covert shifts of visual attention. *Vision Research*, 40(1012):1489 – 1506, 2000.

[101] Jetter, H.-C, Leifert, S, Gerken, J, Schubert, S, and Reiterer, H. Does (multi-)touch aid users' spatial memory and navigation in 'panning' and in 'zooming & panning' UIs? In *Proceedings of the International Working Conference on Advanced Visual Interfaces*, AVI '12, Capri Island, Italy, 2012. ACM, pages 83–90.

[102] Jiang, Y, Song, J, and Rigas, A. High-capacity spatial contextual memory. *Psychonomic Bulletin & Review*, 12(3):524–529, 2005.

[103] Jones, W. P and Dumais, S. T. The spatial metaphor for user interfaces: experimental tests of reference by location versus name. *ACM Transactions on Information Systems*, 4(1):42–63, January 1986.

[104] Jonides, J and Yantis, S. Uniqueness of abrupt visual onset in capturing attention. *Perception & Psychophysics*, 43(4):346–354, 1988.

[105] Jul, S and Furnas, G. W. Critical zones in desert fog: aids to multiscale navigation. In *Proceedings of the 11th annual ACM symposium on User interface software and technology*, UIST '98, San Francisco, California, USA, 1998. ACM, pages 97–106.

[106] Kabay, M. E. Arrogance or efficiency? a discussion of the microsoft office fluent user interface. *Ubiquity*, 2008(March):8:1–8:4, March 2008.

[107] Kaptelinin, V. Item recognition in menu selection: the effect of practice. In *INTERACT '93 and CHI '93 Conference Companion on Human Factors in Computing Systems*, CHI '93, Amsterdam, The Netherlands, 1993. ACM, pages 183–184.

[108] Kaufmann, B and Ahlström, D. Revisiting peephole pointing: a study of target acquisition with a handheld projector. In *Proceedings of the 14th international conference on Human-computer interaction with mobile devices and services*, MobileHCI '12, San Francisco, California, USA, 2012. ACM, pages 211–220.

[109] Kaufmann, B and Ahlström, D. Studying spatial memory and map navigation performance on projector phones with peephole interaction. In *Proceedings of the 2013 ACM annual conference on Human factors in computing systems*, CHI '13, Paris, France, 2013. ACM, pages 3173–3176.

[110] Kearins, J. M. Visual spatial memory in australian aboriginal children of desert regions. *Cognitive Psychology*, 13(3):434 – 460, 1981.

[111] Kelleher, C and Pausch, R. Stencils-based tutorials: Design and evaluation. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '05, Portland, Oregon, USA, 2005. ACM, pages 541–550.

[112] Kiger, J. I. The depth/breadth trade-off in the design of menu-driven user interfaces. *International Journal of Man-Machine Studies*, 20(2):201 – 213, 1984.

[113] Kimura, D. *Sex and cognition*. MIT Press, 2000.

[114] King, D. L, Jones, F. L, Pearlman, R. C, Tishman, A, Felix, C. A, et al. The length of the retention interval, forgetting, and subjective similarity. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, 28(4):660, 2002.

[115] Krendel, E. S and Wodinsky, J. Search in an unstructured visual field. *Journal of the Optical Society of America*, 50(6):562, 1960.

[116] Kurtenbach, G, Fitzmaurice, G. W, Owen, R. N, and Baudel, T. 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, Pittsburgh, Pennsylvania, USA, 1999. ACM, pages 231–237.

[117] Lafreniere, B, Bunt, A, Whissell, J. S, Clarke, C. L. A, and Terry, M. Characterizing large-scale use of a direct manipulation application in the wild. In *Proceedings of Graphics Interface 2010*, GI '10, Ottawa, Ontario, Canada, 2010. Canadian Information Processing Society, pages 11–18.

[118] Lafreniere, B, Bunt, A, Lount, M, Krynicki, F, and Terry, M. A. Adaptablegimp: Designing a socially-adaptable interface. In *Proceedings of the 24th Annual ACM Symposium Adjunct on User Interface Software and Technology*, UIST '11 Adjunct, Santa Barbara, California, USA, 2011. ACM, pages 89–90.

[119] Lam, H, Rensink, R. A, and Munzner, T. Effects of 2d geometric transformations on visual memory. In *Proceedings of the 3rd Symposium on Applied Perception in Graphics and Visualization*, APGV '06, Boston, Massachusetts, 2006. ACM, pages 119–126.

[120] Landauer, T. K and Nachbar, D. W. Selection from alphabetic and numeric menu trees using a touch screen: breadth, depth, and width. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '85, San Francisco, California, USA, 1985. ACM, pages 73–78.

[121] Lane, D. M, Napier, H. A, Peres, S. C, and Sandor, A. 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):133–144, 2005.

[122] Lansdale, M. W. Modeling memory for absolute location. *Psychological Review*, 105(2):351 – 378, 1998.

[123] Lee, D.-S and Yoon, W. C. Quantitative results assessing design issues of selection-supportive menus. *International Journal of Industrial Ergonomics*, 33(1):41–52, January 2004.

[124] Lee, E and Macgregor, J. Minimizing user search time in menu retrieval systems. *Human Factors: The Journal of the Human Factors and Ergonomics Society*, 27(2):157–162, 1985.

[125] Leitheiser, B and Munro, D. An experimental study of the relationship between spatial ability and the learning of a graphical user interface. In *Proceedings of the Inaugural Americas Conference on Information Systems*, 1995, pages 122–124.

[126] Li, F. C. Y, Dearman, D, and Truong, K. N. Virtual shelves: interactions with orientation aware devices. In *Proceedings of the 22nd annual ACM symposium on User interface software and technology*, UIST '09, Victoria, BC, Canada, 2009. ACM, pages 125–128.

[127] Light, L. L and Zelinski, E. M. Memory for spatial information in young and old adults. *Developmental Psychology*, 19(6):901 – 906, 1983.

[128] Lindsay, P and Norman, D. *Human information processing: An introduction to psychology*. Academic Press, 1977.

[129] Logan, G. D. What is learned during automatization? ii. obligatory encoding of spatial location. *Journal of Experimental Psychology: Human Perception and Performance*, 24:1720–1736, 1998.

[130] Logie, R. H. Visuo-spatial processing in working memory. *The Quarterly Journal of Experimental Psychology Section A*, 38(2):229–247, 1986.

[131] Logie, R. H, Zucco, G. M, and Baddeley, A. D. Interference with visual short-term memory. *Acta Psychologica*, 75(1):55 – 74, 1990.

[132] MacGregor, J and Lee, E. Menu search: random or systematic? *International Journal of Man-Machine Studies*, 26(5):627 – 631, 1987.

[133] Mackay, W. E and Fayard, A.-L. Hci, natural science and design: a framework for triangulation across disciplines. In *Proceedings of the 2nd conference on Designing interactive systems: processes, practices, methods, and techniques*, DIS '97, Amsterdam, The Netherlands, 1997. ACM, pages 223–234.

[134] Maguire, E. A and Cipolotti, L. Selective sparing of topographical memory. *Journal of Neurology, Neurosurgery & Psychiatry*, 65(6):903–909, 1998.

[135] Maguire, E. A, Burgess, N, and OKeefe, J. Human spatial navigation: cognitive maps, sexual dimorphism, and neural substrates. *Current Opinion in Neurobiology*, 9(2):171 – 177, 1999.

[136] Malacria, S, Bailly, G, Harrison, J, Cockburn, A, and Gutwin, C. Promoting hotkey use through rehearsal with ExposeHK. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '13, Paris, France, 2013. ACM, pages 573–582.

[137] Mandler, J, Seegmiller, D, and Day, J. On the coding of spatial information. *Memory & Cognition*, 5(1):10–16, 1977.

[138] Matejka, J, Grossman, T, and Fitzmaurice, G. Patina: Dynamic heatmaps for visualizing application usage. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '13, Paris, France, 2013. ACM, pages 3227–3236.

[139] McCarthy, R. A, Evans, J. J, and Hodges, J. R. Topographic amnesia: spatial memory disorder, perceptual dysfunction, or category specific semantic memory impairment? *Journal of Neurology, Neurosurgery & Psychiatry*, 60(3):318–325, 1996.

[140] McCormack, P. D. Coding of spatial information by young and elderly adults. *Journal of Gerontology*, 37(1):80–86, 1982.

[141] McGrath, J. E. Methodology matters: doing research in the behavioral and social sciences. In Baecker, R. M, Grudin, J, Buxton, W. A. S, and

Greenberg, S, editors, *Human-computer interaction*, pages 152–169. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1995.

[142] McGrenere, J, Baecker, R. M, and Booth, K. S. A field evaluation of an adaptable two-interface design for feature-rich software. *ACM Transactions on Computer-Human Interaction*, 14(1), May 2007.

[143] Miller, D. P. The depth/breadth tradeoff in hierarchical computer menus. *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, 25(1):296–300, 1981.

[144] Milner, B. Interhemispheric differences in the localization of psychological processes in man. *British Medical Bulletin*, 1971.

[145] Mitchell, J and Shneiderman, B. Dynamic versus static menus: an exploratory comparison. *SIGCHI Bulletin*, 20(4):33–37, April 1989.

[146] Mou, W, McNamara, T. P, et al. Intrinsic frames of reference in spatial memory. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, 28(1):162–170, 2002.

[147] Mou, W, Biocca, F, Owen, C. B, Tang, A, Xiao, F, and Lim, L. Frames of reference in mobile augmented reality displays. *Journal of Experimental Psychology: Applied*, 10(4):238–244, 2004.

[148] Mou, W, Xiao, C, and McNamara, T. P. Reference directions and reference objects in spatial memory of a briefly viewed layout. *Cognition*, 108(1): 136 – 154, 2008.

[149] Naveh-Benjamin, M. Coding of spatial location information: An automatic process?. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, 13(4):595 – 605, 1987.

[150] Naveh-Benjamin, M. Recognition memory of spatial location information: Another failure to support automaticity. *Memory & Cognition*, 16(5):437–445, 1988.

[151] Nelson, T. O, Chaiklin, S, et al. Immediate memory for spatial location. *Journal of experimental psychology. Human learning and memory*, 6(5): 529, 1980.

[152] Newell, A and Rosenbloom, P. S. Mechanisms of skill acquisition and the law of practice. *Cognitive skills and their acquisition*, pages 1–55, 1981.

[153] Nielsen, C. M, Overgaard, M, Pedersen, M. B, Stage, J, and Stenild, S. It's worth the hassle!: the added value of evaluating the usability of mobile systems in the field. In *Proceedings of the 4th Nordic conference on Human-computer interaction: changing roles*, NordiCHI '06, Oslo, Norway, 2006. ACM, pages 272–280.

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

[155] Nilsen, E. L. Perceptual-motor control in human-computer interaction. Technical Report 37, Ann Arbor, Michigan: The Cognitive Science and Machine Intelligence Laboratory, The University of Michigan, 1996.

[156] Norman, K. L. Spatial visualization–a gateway to computer-based technology. *Journal of Special Education Technology*, 12(3):195–206, 1994.

[157] O'Hara, K and Sellen, A. A comparison of reading paper and on-line documents. In *Proceedings of the ACM SIGCHI Conference on Human factors in computing systems*, CHI '97, Atlanta, Georgia, USA, 1997. ACM, pages 335–342.

[158] O'Hara, K, Sellen, A, and Bentley, R. Supporting memory for spatial location while reading from small displays. In *CHI '99 Extended Abstracts on Human Factors in Computing Systems*, CHI EA '99, Pittsburgh, Pennsylvania, 1999. ACM, pages 220–221.

[159] Pahud, M, Hinckley, K, Iqbal, S, Sellen, A, and Buxton, B. Toward compound navigation tasks on mobiles via spatial manipulation. In *Proceedings of the 15th international conference on Human-computer interaction*

with mobile devices and services, MobileHCI '13, Munich, Germany, 2013. ACM, pages 113–122.

[160] Park, D. C, Puglisi, J. T, and Lutz, R. Spatial memory in older adults: Effects of intentionality. *Journal of Gerontology*, 37(3):330–335, 1982.

[161] Parkhurst, D, Law, K, and Niebur, E. Modeling the role of salience in the allocation of overt visual attention. *Vision Research*, 42(1):107 – 123, 2002.

[162] Pazzaglia, F and Cornoldi, C. The role of distinct components of visuo-spatial working memory in the processing of texts. *Memory*, 7(1):19–41, 1999. PMID: 10645371.

[163] Postma, A and De Haan, E. What was where? memory for object locations. *The Quarterly Journal of Experimental Psychology Section A*, 49(1):178–199, 1996. PMID: 8920102.

[164] Quinn, P, Cockburn, A, and Delamarche, J. Examining the costs of multiple trajectory pointing techniques. *Int. J. Hum.-Comput. Stud.*, 71(4):492–509, April 2013.

[165] Quinn, P and Cockburn, A. The effects of menu parallelism on visual search and selection. In *Proceedings of the ninth conference on Australasian user interface - Volume 76*, AUIC '08, Wollongong, Australia, 2008. Australian Computer Society, Inc., pages 79–84.

[166] Quinn, P, Cockburn, A, Indratmo, and Gutwin, C. An investigation of dynamic landmarking functions. In *Proceedings of the working conference on Advanced visual interfaces*, AVI '08, Napoli, Italy, 2008. ACM, pages 322–325.

[167] Rädle, R, Jetter, H.-C, Butscher, S, and Reiterer, H. The effect of egocentric body movements on users' navigation performance and spatial memory in zoomable user interfaces. In *Proceedings of the 2013 ACM international conference on Interactive tabletops and surfaces*, ITS '13, St. Andrews, Scotland, United Kingdom, 2013. ACM, pages 23–32.

[168] Rayner, K. Eye movements in reading and information processing: 20 years of research. *Psychological bulletin*, 124(3):372, 1998.

[169] Robertson, G, Czerwinski, M, Larson, K, Robbins, D. C, Thiel, D, and van Dantzich, M. Data mountain: using spatial memory for document management. In *Proceedings of the 11th annual ACM symposium on User interface software and technology*, UIST '98, San Francisco, California, USA, 1998. ACM, pages 153–162.

[170] Rothkopf, E. Z. Incidental memory for location of information in text. *Journal of Verbal Learning and Verbal Behavior*, 10(6):608 – 613, 1971.

[171] Sala, S. D, Gray, C, Baddeley, A, Allamano, N, and Wilson, L. Pattern span: a tool for unwelding visuospatial memory. *Neuropsychologia*, 37 (10):1189 – 1199, 1999.

[172] Samp, K. Designing graphical menus for novices and experts: connecting design characteristics with design goals. In *Proceedings of the 2013 ACM annual conference on Human factors in computing systems*, CHI '13, Paris, France, 2013. ACM, pages 3159–3168.

[173] Scarr, J, Cockburn, A, Gutwin, C, and Quinn, P. 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, Vancouver, BC, Canada, 2011. ACM, pages 2741–2750.

[174] Schmidt, R and Lee, T. *Motor Control and Learning: A Behavioral Emphasis*. Human Kinetics, 2011.

[175] Schmidt, R. A. Motor learning & performance: From principles to practice. 1991.

[176] Sears, A and Shneiderman, B. Split menus: effectively using selection frequency to organize menus. *ACM Transactions on Computer-Human Interaction*, 1(1):27–51, March 1994.

[177] Sears, A, Jacko, J. A, Chu, J, and Moro, F. The role of visual search in the design of effective soft keyboards. *Behaviour & Information Technology*, 20(3):159–166, 2001.

[178] Sharps, M. J and Gollin, E. S. Memory for object locations in young and elderly adults. *Journal of Gerontology*, 42(3):336–341, 1987.

[179] Shelton, A. L and McNamara, T. P. Multiple views of spatial memory. *Psychonomic Bulletin & Review*, 4(1):102–106, 1997.

[180] Shelton, A. L, McNamara, T. P, et al. Systems of spatial reference in human memory. *Cognitive Psychology*, 43(4):274–310, 2001.

[181] Shneiderman, B, Plaisant, C, Cohen, M, and Jacobs, S. *Designing the User Interface: Strategies for Effective Human-Computer Interaction*. Addison-Wesley Publishing Company, USA, 5th edition, 2009.

[182] Siegel, A. W and White, S. H. The development of spatial representations of large-scale environments. volume 10 of *Advances in Child Development and Behavior*, pages 9 – 55. JAI, 1975.

[183] Skopik, A and Gutwin, C. Improving revisitation in fisheye views with visit wear. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '05, Portland, Oregon, USA, 2005. ACM, pages 771–780.

[184] Smith, R. B and Taivalsaari, A. Generalized and stationary scrolling. In *Proceedings of the 12th annual ACM symposium on User interface software and technology*, UIST '99, Asheville, North Carolina, USA, 1999. ACM, pages 1–9.

[185] Snowberry, K, Parkinson, S. R, and Sisson, N. Computer display menus. *Ergonomics*, 26(7):699–712, 1983.

[186] Somberg, B. L. A comparison of rule-based and positionally constant arrangements of computer menu items. In *Proceedings of the SIGCHI/GI*

Conference on Human Factors in Computing Systems and Graphics Interface, CHI '87, Toronto, Ontario, Canada, 1987. ACM, pages 255–260.

[187] Sorrows, M and Hirtle, S. The nature of landmarks for real and electronic spaces. In Freksa, C and Mark, D, editors, *Spatial Information Theory. Cognitive and Computational Foundations of Geographic Information Science*, volume 1661 of *Lecture Notes in Computer Science*, pages 37–50. Springer Berlin Heidelberg, 1999.

[188] Soukoreff, R. W and Mackenzie, I. S. Theoretical upper and lower bounds on typing speed using a stylus and a soft keyboard. *Behaviour & Information Technology*, 14(6):370–379, 1995.

[189] Steenhuis, R. E and Goodale, M. A. The effects of time and distance on accuracy of target-directed locomotion: does an accurate short-term memory for spatial location exist? *Journal of Motor Behavior*, 20(4):399–415, 1988.

[190] Tak, S and Cockburn, A. Enhanced spatial stability with Hilbert and Moore treemaps. *Visualization and Computer Graphics, IEEE Transactions on*, 19 (1):141–148, 2013.

[191] Tak, S, Cockburn, A, Humm, K, Ahlström, D, Gutwin, C, and Scarr, J. Improving window switching interfaces. In *Proceedings of the 12th IFIP TC 13 International Conference on Human-Computer Interaction: Part II*, INTERACT '09, Uppsala, Sweden, 2009. Springer-Verlag, pages 187–200.

[192] Tak, S, Scarr, J, Gutwin, C, and Cockburn, A. Supporting window switching with spatially consistent thumbnail zones: design and evaluation. In *Proceedings of the 13th IFIP TC 13 international conference on Human-computer interaction - Volume Part I*, INTERACT'11, Lisbon, Portugal, 2011. Springer-Verlag, pages 331–347.

[193] Tan, D. S, Stefanucci, J. K, Proffitt, D. R, and Pausch, R. The infocockpit: providing location and place to aid human memory. In *Proceedings of the*

2001 workshop on Perceptive user interfaces, PUI '01, Orlando, Florida, 2001. ACM, pages 1–4.

[194] Tan, D. S, Pausch, R, Stefanucci, J. K, and Proffitt, D. R. Kinesthetic cues aid spatial memory. In *CHI '02 Extended Abstracts on Human Factors in Computing Systems*, CHI EA '02, Minneapolis, Minnesota, USA, 2002. ACM, pages 806–807.

[195] Tan, D. S, Gergle, D, Scupelli, P, and Pausch, R. Physically large displays improve performance on spatial tasks. *ACM Transactions on Computer-Human Interaction*, 13(1):71–99, March 2006.

[196] Tavanti, M and Lind, M. 2d vs 3d, implications on spatial memory. In *Information Visualization, 2001. INFOVIS 2001. IEEE Symposium on*, 2001, pages 139–145.

[197] Thomson, J. A et al. Is continuous visual monitoring necessary in visually guided locomotion. *Journal of Experimental Psychology: Human Perception and Performance*, 9(3):427–443, 1983.

[198] Tlauka, M, Donaldson, P. K, and Wilson, D. Forgetting in spatial memories acquired in a virtual environment. *Applied Cognitive Psychology*, 22(1): 69–84, 2008.

[199] Treisman, A. Perceptual grouping and attention in visual search for features and for objects. *Journal of Experimental Psychology: Human Perception and Performance*, 8(2):194, 1982.

[200] Treisman, A. M and Gelade, G. A feature-integration theory of attention. *Cognitive Psychology*, 12(1):97 – 136, 1980.

[201] Tsang, M, Fitzmaurice, G. W, Kurtenbach, G, Khan, A, and Buxton, B. Boom chameleon: simultaneous capture of 3d viewpoint, voice and gesture annotations on a spatially-aware display. In *Proceedings of the 15th annual ACM symposium on User interface software and technology*, UIST '02, Paris, France, 2002. ACM, pages 111–120.

[202] Tu, Y and Shen, H.-W. Visualizing changes of hierarchical data using treemaps. *Visualization and Computer Graphics, IEEE Transactions on*, 13(6):1286–1293, 2007.

[203] TVERSKY, B, MORRISON, J. B, and BETRANCOURT, M. Animation: can it facilitate? *International Journal of Human-Computer Studies*, 57(4): 247 – 262, 2002.

[204] Underwood, B. J. Interference and forgetting. *Psychological Review*, 64 (1):49 – 60, 1957.

[205] van Asselen, M, Fritschy, E, and Postma, A. The influence of intentional and incidental learning on acquiring spatial knowledge during navigation. *Psychological Research*, 70:151–156, 2006.

[206] von Wright, J, Gebhard, P, and Karttunen, M. A developmental study of the recall of spatial location. *Journal of Experimental Child Psychology*, 20(1):181 – 190, 1975.

[207] Watson, N. V and Kimura, D. Nontrivial sex differences in throwing and intercepting: Relation to psychometrically-defined spatial functions. *Personality and Individual Differences*, 12(5):375 – 385, 1991.

[208] Wertheimer, M. Laws of organization in perceptual forms. 1938.

[209] Wixted, J. T. The psychology and neuroscience of forgetting. *Annual review of psychology*, 55:235–269, 2004.

[210] Wobbrock, J. O and Myers, B. A. Enabling devices, empowering people: The design and evaluation of trackball edgewrite. *Disability and Rehabilitation: Assistive Technology*, 3(1-2):35–56, 2008.

[211] Yantis, S and Jonides, J. Abrupt visual onsets and selective attention: Evidence from visual search. *Journal of Experimental Psychology: Human perception and performance*, 10(5):601–621, 1984.

[212] Yates, F. *The Art of Memory*. A Phoenix book. University of Chicago Press, 1966.

[213] Yee, K.-P. Peephole displays: pen interaction on spatially aware handheld computers. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '03, Ft. Lauderdale, Florida, USA, 2003. ACM, pages 1–8.

[214] Zanella, A, Carpendale, M. S. T, and Rounding, M. On the effects of viewing cues in comprehending distortions. In *Proceedings of the second Nordic conference on Human-computer interaction*, NordiCHI '02, Aarhus, Denmark, 2002. ACM, pages 119–128.

[215] Zellweger, P. T, Mackinlay, J. D, Good, L, Stefik, M, and Baudisch, P. City lights: contextual views in minimal space. In *CHI '03 Extended Abstracts on Human Factors in Computing Systems*, CHI EA '03, Ft. Lauderdale, Florida, USA, 2003. ACM, pages 838–839.

Appendices

Appendix A

Publication: Skillometers: Reflective Widgets that Motivate and Help Users to Improve Performance

Skillometers: Reflective Widgets that Motivate and Help Users to Improve Performance

Sylvain Malacria[1] **Joey Scarr**[1] **Andy Cockburn**[1] **Carl Gutwin**[2] **Tovi Grossman**[3]
[1]University of Canterbury, Christchurch, New Zealand
[2]Department of Computer Science, University of Saskatchewan, Saskatoon, Saskatchewan, Canada
[3]Autodesk Research, Toronto, Ontario, Canada
sylvain@malacria.fr, {joey, andy}@cosc.canterbury.ac.nz, gutwin@cs.usask.ca, tovi.grossman@autodesk.com

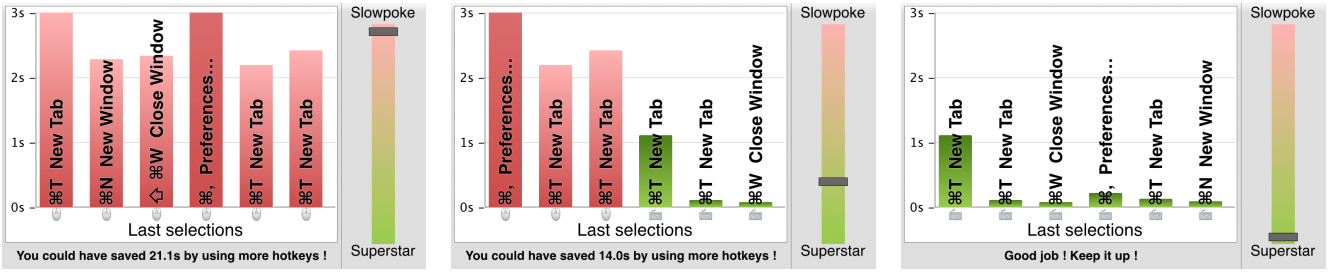


Figure 1. Three snapshots of a skillometer designed to encourage hotkey use. At first (left), the user mostly relies on mouse selections for activating commands. The skillometer indicates that he could save time by using hotkeys, and displays the appropriate hotkey bindings. As he begins to increase his hotkey use (center), the skillometer shows the benefits of the switch, encouraging him to use more hotkeys (right).

ABSTRACT

Applications typically provide ways for expert users to increase their performance, such as keyboard shortcuts or customization, but these facilities are frequently ignored. To help address this problem, we introduce *skillometers* – lightweight displays that visualize the benefits available through practicing, adopting a better technique, or switching to a faster mode of interaction. We present a general framework for skillometer design, then discuss the design and implementation of a real-world skillometer intended to increase hotkey use. A controlled experiment shows that our skillometer successfully encourages earlier and faster learning of hotkeys. Finally, we discuss general lessons for future development and deployment of skillometers.

Author Keywords

Expertise; hotkeys; learning; reflection.

ACM Classification Keywords

H.5.2. Information Interfaces and Presentation: User Interfaces. - Graphical user interfaces.

General Terms

Human Factors; Design; Measurement.

UIST'13, October 6–9, 2013, St. Andrews, United Kingdom.

ACM 978-1-4503-2271-3/13/10...$15.00.
http://dx.doi.org/10.1145/2501988.2501996

INTRODUCTION

In modern applications, users have many opportunities to enhance their productivity. They can alter their environment (e.g., customizing the interface to suit their workflows and bring frequently-used commands close to hand), or they can alter themselves by learning new skills and techniques for the application. These new techniques can dramatically improve performance, but a major problem in many systems is that users ignore these opportunities [23, 27]. For example, many users opt to "satisfice" [38] and stick with the interaction procedures they learnt as novices, even when they are aware of the expert features [12].

This behavior is not unreasonable. Adopting a new mechanism or strategy could require a significant investment of time and effort, possibly outweighing any efficiency gains; if the user has no way of estimating these costs and benefits, then they should not be expected to switch. Critically, it is the user's *perception* of the new mechanism that matters [35], rather than its actual properties – therefore, if we could inform the user about the benefits of adopting a new mechanism or strategy, they might be more inclined to do so.

In this paper, we present and analyse the design potential of using *skillometers* as a solution to these problems. Skillometers are widgets designed to accelerate the development of UI expertise by allowing the user to visualize their past and potential future performance. While the term 'skillometer' has been used in the domain of intelligent tutoring systems to reflect the system's model of the user's understanding (e.g., [4, 5]), we are unaware of their use for interface performance, except for Linton and Schaefer's proposal to use a "Skill-O-Meter" to visualize command vocabulary [26]. Other related systems are reviewed later in the paper. The specific skil-

lometer implemented and evaluated in this paper, shown in Figure 1, displays a visualization of a user's interaction efficiency (in this case, with menus and keyboard shortcuts), and suggests ways in which efficiency could be improved.

This paper makes three specific contributions. First, we present a general skillometer framework which can be used to inform the design of future skillometers. Second, we discuss the design of HotkeySkillometer, a skillometer for encouraging hotkey use, and describe its implementation. Finally, we present the results of a controlled experiment, where participants performed tightly constrained tasks in Apple Keynote. Results demonstrate that the HotkeySkillometer motivates users to increase their use of hotkeys. We conclude by discussing the lessons learned and issues of generalisation.

SKILLOMETER DEFINITION

Reflective interfaces, which inform the users about their own behavior, have been well-studied in the area of tutoring systems (e.g. [5, 24]), which are specifically designed to help the user develop skills and knowledge. In the context of interaction, Linton and Schaefer [26] proposed a reflective visualization called a 'Skill-O-Meter', which displays the user's command vocabulary compared to others in their peer group. However, the effect of this feedback on user behavior was not evaluated. More recently, Bateman et al. [8] developed a 'Search Dashboard' that gave users information about their search habits, including comparison of their behaviors to those of typical and expert users. By displaying information about expert behavior, the Search Dashboard was able to broaden users' knowledge of search engine features.

In this paper, we define a *skillometer* as a reflective widget that motivates and helps the user increase their level of performance with an interface. Importantly, as reflective widgets, skillometers support the user in understanding meta-level aspects of their own interaction. Typically, the meta-level aspect of interaction revealed by a skillometer will concern the user's level of performance, although others may be revealed, such as the user's task strategy or interaction modalities used to select commands (examples follow later in the paper).

Under this definition, a tooltip that displays a hotkey binding would not be considered to be a skillometer because although it assists the user in increasing their level of performance (by helping the user learn the hotkey), it does not facilitate understanding of performance at a higher level. In contrast, the widget shown in Figure 1 is a skillometer because it displays the user's level of performance (a meta-level aspect of interaction) as well as showing hotkey bindings that might be used to complete pointer-based selections (shown in red).

TARGET DOMAINS OF USER PERFORMANCE

Skillometers can be deployed to improve four different target domains of user performance with interfaces: *intramodal*, *intermodal*, *vocabulary extension*, and *strategic*. Each of these domains is described with respect to potential skillometer design and prior work in upcoming subsections. In brief, *intramodal* improvement addresses the rapidity of skill acquisition within one particular interactive method for one particular function (e.g., assisting the user to improve their performance with a novel pointing device); *intermodal* improvement addresses ways to assist users to switch to faster methods for accessing a particular function (e.g., switching from cursor-based pointing to hotkeys); *vocabulary extension* methods focus on broadening the user's knowledge of the range of functions available through the interface; and *strategic* methods address higher level issues of how users combine interface functions to accomplish tasks.

Domain 1: Intramodal Improvement

Human skill acquisition has been shown to reliably follow a 'power law of practice' [31] across a wide range of activities, including interaction with computer systems. The law of practice shows that performance improves quickly at first, gradually leveling off with experience. In many interaction contexts, such as cursor-based selection of menu items, this effect is easily explained: users who are unfamiliar with the interface must initially rely on visual search to identify likely targets, but can gradually use faster methods for interaction based on spatial, muscular and procedural memory. After extensive experience with any particular interaction modality, users will approach a performance ceiling representing the limit of the user's capability for that modality.

While performance improvement within any particular interaction modality will follow a power law of practice, research in the psychomotor literature clearly shows that different forms of practice and training can yield different rates of skill acquisition – see [36] for an extensive review. For example, there are reliable experimental effects for various practice manipulations, including the timing and distribution of practice, the mental effort associated with processing trained material, the specificity of the training material, and various feedback effects.

Accelerating the rate of skill acquisition is one opportunity for skillometers to improve performance. For example, some psychology literature suggests that *knowledge of performance* can improve skill acquisition (although results can vary [36]), and this effect might be employed by a skillometer. A touch-typing skillometer, for instance, might show a real-time line chart of the user's word-per-minute rate (Figure 2a). Additionally, many psychology and sports studies shows that appropriate goal-setting can improve skill acquisition (again, see [36]). This result might be incorporated into the skillometer by continually showing the user's peak typing rate as another line on the chart above their current rate.

Domain 2: Intermodal Improvement

Most interfaces support more than one interaction modality for accessing the same function: for example, the 'Bold' function might be activated by selecting it from a menu, clicking a toolbar button, choosing it in a dialog box, or pressing Ctrl+B. The rate of skill acquisition for each independent modality is the concern of *intramodal* improvement. *Intermodal* improvement, in contrast, concerns the transition from slower modalities to faster ones that ultimately offer a higher performance ceiling.

Many studies have observed that expert interface modalities are often unused or only lightly used. Carroll [12] attributed

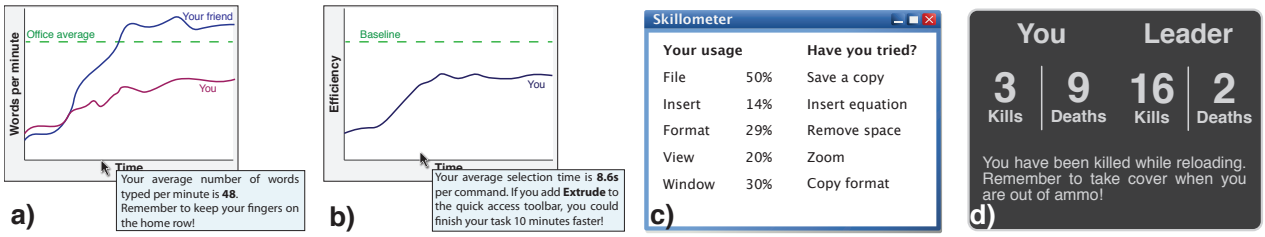


Figure 2. Concept sketches of skillometers for a) typing speed (intramodal), b) interface customization (intermodal), c) command awareness (vocabulary extension) and d) video game training (strategic).

the effect to the *paradox of the active user*, noting that users tend to be too engaged in their primary task to put cognitive effort into learning more efficient ways to accomplish that task. This behavior is an example of a wider human phenomenon called *satisficing* [38], where people make do with sufficient, but suboptimal, strategies due to limited cognitive resources: typical users do not have the necessary information to determine whether learning hotkeys will be a worthwhile investment, nor can they spare the substantial effort required to acquire that information. In related work, Scarr et al. [35] further examined users' failure to make a transition to higher-performance expert modalities, noting that a temporary 'performance dip' while gaining familiarity with a new modality can deter users from switching to a new method.

Several factors act as deterrents or barriers to users switching to modalities that can offer a higher performance ceiling, including the user's lack of awareness of other modalities, their perception of performance with other modalities, and their lack of motivation to use them. Each of these is described below, with particular reference to the opportunities for skillometers.

Awareness of Other Modalities

There are many ways to increase awareness of advanced interaction modalities. For example, keyboard shortcuts are sometimes displayed to the right of menu labels, or in tooltips that appear after a dwell. However, Grossman et al.'s results suggest that even when shortcuts are displayed in dropdown menus, they are commonly ignored by users [20]. This may be attributable to the fact that the shortcut is not displayed until the user has already done most of the work for selecting the command with the mouse, so they have no incentive to learn the hotkey. Malacria et al. [28] and Tak et al. [39] both developed successful systems that provide "feedforward" information about hotkeys when a modifier key (e.g. Ctrl) is held down, displaying hotkey mappings when they are needed rather than after the fact, with both systems increasing hotkey use. Krisler and Alterman showed improved hotkey learning with their HotKeyCoach [22], which displayed a pop-up window that needed to be explicitly dismissed every time the user selected a command with the mouse. However, explicit awareness techniques often come at the cost of undue distraction, and one of the goals of our skillometer is to minimise interference and disruption of normal interaction (explicit dialog boxes clearly violate this goal). Scarr et al.'s "calm notification" [35] technique provides awareness in a less intrusive fashion by using an ambient dialog in the corner of the screen that does not require user response and that automatically fades away if ignored.

The visual presentation of a skillometer widget can incorporate awareness information about appropriate expert modalities. Using a skillometer as the location for this information has two important advantages over other methods: first, it provides a consistent location for feedback on higher-performance methods for completing recent actions; second, the skillometer can show performance over time, allowing users to compare past and potential future performance (e.g., the skillometer in Figure 1 shows information about the last six commands).

Perception of Performance with Other Modalities

In their framework of interface expertise, Scarr et al. [35] note that the user's *perception* of their potential performance in a modality (both over the short and the long term) is a critical factor influencing their decision of whether to switch. However, a user's perception of future performance is often unreliable [13]. In the context of command selection, studies have consistently shown that keyboard shortcuts offer a higher performance ceiling than mouse selection [23, 32], yet Tak et al. found that some participants failed to use known hotkeys because they believed the toolbar buttons to be faster [39].

Appropriate skillometer design may assist with communicating accurate perceptions of the relative potential and actual performance gains of using different modalities. For example, a skillometer might show the actual time taken to select the 'bold' command in a ribbon or menu (calculating time from completing a text selection and selecting the item) and also show the estimated keystroke time for the associated hotkey. Alternatively, time estimates might be aggregated and/or projected to future total time savings, based on the frequency with which particular commands have been used in the past (e.g., the tooltip in Figure 2b). Such information might assist the user in making rational judgements concerning the costs and benefits of changing modality.

Lack of Motivation

The user's motivation for improving performance is another important factor. There are many elements of motivation at play in a user's decision to learn new skills and techniques, and skillometer design can make use of both intrinsic and extrinsic elements. First, intrinsic motivation concerns the user's own interest in improving, and here it is important for the skillometer to clearly visualize the potential for future improvement. However, designers must recognize that

the user's performance goal may depend upon the context – for example, a user browsing the Internet at home may be less concerned with fast command selection than an office worker who carries out the same tasks throughout the day. Interestingly, however, Tak et al. [39] found that increased time pressure did not lead to increased hotkey use, suggesting that workload alone is not sufficient to encourage adoption of more efficient methods (reflecting Carroll's paradox).

Extrinsic motivation can also be used in skillometer design – in particular, there are interesting opportunities for skillometers to use social factors, social learning, and gamification (discussed later) as motivation. For example, recent results from Banovic et al. [7] showed that social influence was important for encouraging tool palette customization. Similarly, Peres et al. [33] found that users were more likely to use keyboard shortcuts if they worked with others who did so, and Bateman et al. [8] found that having knowledge of expert search engine use led people to change their own behaviour.

Domain 3: Vocabulary Extension

Many interfaces allow access to extensive functionality, but users are often unaware of functions that may assist their performance (see [30] for a review of the problem). For example, in a photo editing suite a user may be familiar with a tool for painting filled circles and may use it to remove red-eye from photographs, unaware that a more effective dedicated red-eye removal function is available. Existing interface mechanisms such as 'tip of the day' can assist in extending the user's vocabulary of commands, but the information they provide is typically unconnected with the user's current task, and may therefore be viewed as an unwanted distraction.

The conceptual deterrents and barriers to vocabulary extension are similar to those described for *intermodal improvement* described above. The opportunities for skillometers to assist with overcoming these deterrents and barriers are also similar to those described for intermodal improvement, although there are additional challenges in determining the functional relationships between groups of commands. While it is straightforward for a skillometer to determine and display intermodal relationships (such as the availability of a hotkey when a menu item is selected by pointing), determining functional relationships is likely to benefit from some semantic knowledge of the user's task objective. For example, a skillometer could only represent the efficiency of the 'red-eye' example above if it correctly identified that the user's manipulation of the circle painting tool was directed at that specific task rather than some other activity.

However, skillometers are not restricted to displaying information about task completion times. Rather, they might be used to promote reflection about other meta-level aspects of interaction, such as the range of commands used. In such a deployment, a skillometer might depict the set of commands that the user employs, grounded by a depiction of the set of commands used by other users – Linton's 'Skill-O-Meter' proposal [26] and Bateman's 'Search Dashboard' [8] are examples. Furthermore, the skillometer might incorporate information about the frequency of use of commands among a community of users, thereby alerting a user to unknown commands that others find useful (e.g., Figure 2c). CommunityCommands [30] and Patina [29] were directed at these objectives, with Patina's interface representation using a heatmap stencil overlay on top of the user interface to show the frequency of command use among a community of users.

Domain 4: Strategic Improvement

Strategic improvement involves helping the user to optimize their sequence of actions – i.e., choosing a better *strategy* to complete the task. While vocabulary extension often facilitates strategic improvement (e.g., invoking 'Align Left' is a better strategy than using drag-and-drop to manually left-align items), users may need more guidance than simply providing broader awareness of application functionality. We refer the reader to Bhavnani et al. [10] for a detailed analysis of strategic aspects of computer use, but we use one of their examples to illustrate interaction strategies. Consider a user interacting with a drawing package to create a series of identical arched windows. A novice user might use a 'sequence-by-operation' strategy, first drawing all of the arcs for the window arches (arc operation), then drawing all of the vertical lines for the window sides (line operation), and finally drawing the horizontal lines for the window bases. An experienced user, in contrast, might use a 'detail-aggregate-manipulate' strategy, first drawing the arc and lines for one window (detail), then grouping the lines (aggregate), and finally copying and pasting multiple replicates (manipulate).

In theory, a skillometer targeted at strategic improvement would detect inefficient interaction strategies and suggest better ones. In general, detecting suboptimal user interaction strategies is a difficult problem. However, in some applications (particularly games), there are points where it is easy to tell that the user has performed poorly. For example, a strategic skillometer for a first-person shooter game might analyse players' deaths and give targeted advice such as 'remember to take cover while reloading' or 'keep your movements unpredictable' (Figure 2d).

SKILLOMETER INTERFACE: GOALS AND MECHANISMS

As discussed above, skillometers may be targeted at any of four different domains of user interface performance. However, regardless of the target domain, there are four goals that apply to the user interface of any skillometer, as follows.

Goal 1: Visualize the Possibility for Improvement

A skillometer interface should clearly communicate to the user that their performance can be improved. This can mean different things, depending on the purpose of the skillometer. An *intramodal* skillometer might display a graph of the user's performance compared to a theoretical optimum. For an *intermodal* skillometer, this means making the user aware of different modalities and their benefits – for example, displaying the user's performance compared to their performance if they used hotkeys. Skillometers aimed at *vocabulary extension* and *strategic improvement* might suggest new functionality and interaction strategies to the user, and demonstrate how these might improve efficiency or quality of work.

Goal 2: Provide Specific Details on How to Improve

As well as making the user aware of the possibility of improving their performance, a skillometer should also provide specific guidance to the user about how that improvement could take place. For example, an intermodal skillometer would provide details of how to perform commands with a better modality, such as showing the hotkey bindings for each command performed.

Goal 3: Motivate the User to Improve

Motivation can be provided by a skillometer in several different ways. A skillometer's visualization should stimulate the user's intrinsic motivation: as discussed in Goal 1, it should enable the user to quantify the benefits of improvement. However, there are additional methods available to motivate and persuade users, discussed below.

Social Influence and Competition

As mentioned earlier, social influence can be an important extrinsic motivation for changing user behavior (e.g., [7, 8, 33]). A skillometer could make use of this by comparing the user's performance, modality use, command knowledge, or interaction strategies to others in their peer group. For example, a touch-typing skillometer could display extra lines on the graph representing the typing speeds of co-workers. However, care must be taken, as some users may react negatively to unfavorable comparison with others.

Exaggeration of Benefits

Since the point of a skillometer is to persuade users to change their behavior, one option would be to exaggerate the advantages of doing so. This is a case of *benevolent deception* [3], where an interface lies in order to provide some benefit to the user. As with competition, care needs to be taken with this approach to avoid negative effects – if the deception is too obvious, the user will lose trust in the system and stop attending to the feedback.

Gamification

Gamification is a broad research topic, and we refer readers to [14] and [21] for a review of gamification techniques. In brief, game elements have been demonstrated to enhance the effectiveness of training with software systems (e.g., [15, 25]), and gamification has been deployed to assist learning of commercial software interfaces (e.g., Microsoft's Ribbon Hero [2]). Gamified interactive components could be used as part of a skillometer, such as scoreboards, achievements, and progression-based elements equivalent to game levels.

Goal 4: Minimize Disruption to Normal Interaction

While providing feedback on user performance can be beneficial, it has also been shown to degrade learning if given too frequently [34]. A skillometer should therefore aim to provide information when it is most useful to the user, and minimize disruption of the user's normal workflow. Research by Bødker [11] suggests that there are two types of task interruption: *breakdowns* and *focus-shifts*. Breakdowns result in severe disruption and force the user's attention to a new activity, while focus-shifts cause only a brief attention switch and have a minimal effect. Systems resulting in a breakdown, such as Microsoft's *Clippy*, have been shown to be quickly abandoned after a few untimely interruptions [37].

Ideally, skillometer feedback would be transient, unintrusive, and require no user response, eliciting a focus-shift rather than a breakdown from the user. However, this presents a difficult trade-off: in order for a skillometer to counteract Carroll's *active user paradox*, a certain amount of disruption is necessary to draw attention to the skillometer.

Manipulating the *locus of control* is one way that a skillometer could avoid this tradeoff. Push notifications, such as a window popping up on screen, can be distracting and interrupt the user's workflow. However, if the locus of control rests with the user, they can choose to request the feedback at a time that suits them, and ignore it when they are busy. The risk of this approach is that the user may never request the feedback – so a possible compromise is a small ambient display in the corner of the screen, which constantly displays a performance overview and provides more detailed information and guidance only on demand.

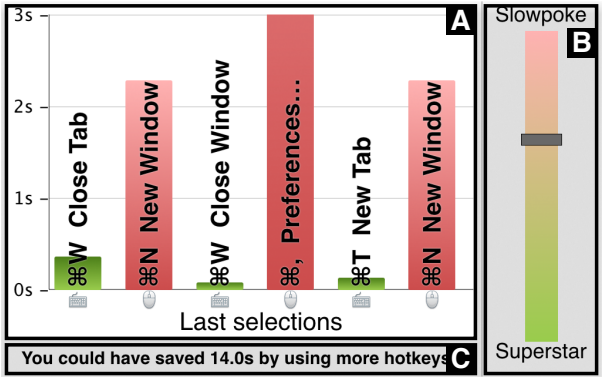


Figure 3. The visual design of HotkeySkillometer.

A SKILLOMETER FOR PROMOTING HOTKEY USE

To gain practical experience with the challenges of designing and implementing skillometers, and to gain initial insights into the validity using skillometers to assist expertise development, we constructed and evaluated a system called HotkeySkillometer. The system, shown in Figure 3, is directed at target domain #2, *intermodal improvement* – it aims to motivate and assist users in transitioning from mouse-driven interaction methods (menu and toolbar selections) to hotkeys [20, 28, 6]. We chose to examine hotkey intermodal improvement because hotkeys are widely available in commercial software, yet they are known to be lightly used [23].

Overview of HotkeySkillometer

HotkeySkillometer monitors the user's interaction with graphical desktop applications, analysing and depicting their level of performance, and displaying alternative and faster hotkey methods for executing commands whenever they are available. Its display consists of three parts, labelled A, B, and C in Figure 3. Part A (Fig. 3, left) uses a bar chart to show the time taken to select each of the six most recently used commands (the Implementation section below details how selection times are calculated to determine each bar's height).

When a new command is selected, an animation slides a new bar item into view, scrolling all other items leftwards. Each bar is coloured dependent on the selection modality (red for mouse, green for hotkey), and an icon below each bar also depicts the modality. Additionally, each bar is overlaid with text showing the name of the command and its hotkey.

Part B (Fig. 3, right) is a meter that grades the user's performance based on the modality used for the last 6 command selections – if all selections were made with hotkeys, the slider will be at the 'Superstar' end; if all used the mouse, it will be at the 'Slowpoke' end. The value of the slider is determined by a weighted interpolation, where more recently selected commands have more weighting – this gives the bar a more noticeable 'swing' toward the positive end each time a hotkey is used, or the negative end with each mouse selection.

Part C (Fig. 3, bottom) shows motivational text advising the user that hotkey selections are faster. It also quantifies, across the past 6 selections, how much time could have been saved if the selections were performed with hotkeys instead (based on a KLM-derived value of 200ms for each hotkey selection).

By default, HotkeySkillometer is a transient window located at the bottom left (or right, based on user preference) of the display. As soon as the user selects a command or presses the Command key, the window fades in and remains until the user stops selecting commands. When the user mouses over the window, it fades out revealing the content beneath it. While the mouse-over fading enables users to interact with underlying content, it also prohibits intentional interaction with components within the skillometer – this design choice is appropriate for HotkeySkillometer, but will be inappropriate for other designs where the user needs to be able to explicitly request information associated with skillometer components (e.g., the roll-over tips shown in Figure 2a).

With respect to the interface goals, HotkeySkillometer supports *Goal 1: Visualize the possibility for improvement* and *Goal 3: Motivate the user to improve* in both the height of mouse-based bar chart items (suggesting poor performance) and through the performance meter, which quickly moves towards the 'Slowpoke' end following mouse selections. It supports *Goal 2: Provide specific details on how to improve* in Part C's text and by the hotkey overlay on the bar chart. *Goal 4: Minimize disruption to normal interaction* is supported by using a transient window that only appears after each command selection and which quickly fades out with inactivity.

IMPLEMENTATION OF HOTKEYSKILLOMETER

The implementation of HotkeySkillometer requires three fundamental building blocks. First, it has to be able to monitor and detect command selections. Second, it has to determine which modality (e.g., mouse or hotkey) was used to select the command. Third, it has to calculate and display the user's performance. Our implementation, HotkeySkillometer, is a Mac OS X application, developed using Objective-C and the Cocoa framework. It uses Apple's Accessibility API [1] to browse and access GUI control elements; all Mac OS X applications implementing this interface can therefore be observed by our skillometer. Details of how HotkeySkillometer achieves each of the three building blocks follow.

Monitoring and Detecting Command Selections

HotkeySkillometer uses two separate components for menus and toolbar selections.

Menu Items

When an application first receives focus, HotkeySkillometer dynamically adds global action listeners to all the menu items located within an application's menu bar and context menus. These action listeners are triggered as soon as the user selects a command (either by clicking on the menu item, or using its associated hotkey), and the listener is provided with all the accessibility attributes of the menu item, which includes the command name and keyboard shortcut.

Toolbar Buttons

Handling toolbar selections is more complicated, since Apple does not provide a global listener for toolbar buttons. Unlike the menu bar, toolbar buttons are often custom widgets that do not implement the Accessibility API, which complicates identifying widgets and accessing any associated hotkey.

To circumvent this limitation, HotkeySkillometer uses a hard-coded description of the mapping between toolbar buttons and their equivalent menu items for each of the client interfaces it monitors. It then uses a combination of a global mouse event listener (that intercepts mouse-up actions) and the Accessibility API to identify the widget located beneath the mouse cursor and thus determine which command was executed. An alternative approach, not used in HotkeySkillometer, would use code injections for extending existing method of the applications at runtime [16]. However, this method would be limited to Cocoa applications and would require a plugin for each monitored application.

Determining the Modality Used

Since toolbar selections use a different detection mechanism to menus, they are easily distinguished. However, for menu items that also possess hotkeys, the same action listener is called for both pointer and hotkey selections. To discriminate between selection modalities, HotkeySkillometer listens to global mouse and keyboard events and checks whether the key corresponding to the keyboard shortcut character was pressed immediately before command selection. If so, the command is recorded as a hotkey selection.

Calculating User Performance

HotkeySkillometer measures pointing time using an Objective-C implementation of Evans and Wobbrock's Input Observer [18], which assumes that a user's current pointing action began with the most recent ballistic movement. The time to select a toolbar item therefore begins at the user's most recent ballistic motion, and ends when the toolbar button is clicked. Menu item selection time is calculated as the sum of the time taken to point to the top-level menu bar item, and the time spent in the menu (and submenus) prior to command selection. For context menus, selection time is simply the time spent in the menu prior to selection.

For keyboard shortcut selections, HotkeySkillometer simply measures the time between the first modifier keypress (or the last command selection in the case of consecutive keyboard shortcut activations) and the command selection time.

Note that both of these measures exclude the time for mental preparation and for precursor movements such as homing the hand to the mouse or key. Consequently, both measures represent the lower-bound of the selection time.

EXPERIMENTAL STUDY

We performed a lab experiment using HotkeySkillometer to gain initial insights into the validity of the skillometer approach to performance improvement. The experimental tasks were conducted using Apple's Keynote software in a between-subjects design, with half of the participants having access to HotkeySkillometer and the other half not.

Participants and Apparatus

24 participants were recruited from a local university (20 male, 4 female). The experiment was performed on an Apple Macbook Pro running Mac OS 10.8. Participants were provided with a standard mouse and QWERTY keyboard, and a 24" screen running at 1920×1080 pixels.

Since Keynote ordinarily lacks keyboard shortcuts for many commands, we used the Mac OS System Preferences to add custom hotkeys for all of the commands used in the experiment. Each shortcut was composed of a single modifier key (Command) followed by one of the 18 alphabetic keys from the left part of the keyboard. No hotkey letter was the same as the first or last letter of the command name.

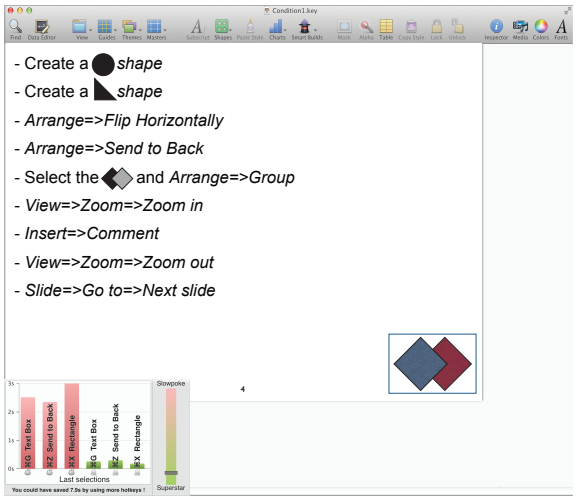


Figure 4. The initial state of a slide used in the experiment. Participants performed tasks on the slide surface, and the instructions were included as a background image. The skillometer appeared at bottom left.

Procedure

Participants were pseudo-randomly assigned to either the *skillometer* or *control* condition (*control* having no access to HotkeySkillometer). Participants were introduced to the Keynote software and to the experimental procedure – they were instructed that they would be completing 'a repetitive series of tasks'. In the *skillometer* condition, participants were additionally given a brief description of the skillometer, explaining the visual presentation of parts A, B, and C shown in Figure 3. To reduce the risk of introducing bias, the skillometer instructions did not mention its overall goal. Participants in both conditions were informed that hotkeys existed for all commands (with labels shown next to menu items). Finally, they were instructed to complete tasks as accurately and quickly as possible.

Each participant performed eight formatting commands on a Keynote slide, then selected a 'next slide' command. They repeated the same eight commands (and next slide) ten times, giving a total of $10 \times 8 = 80$ logged command selections. Instructions for the eight formatting commands were presented in a Keynote slide (Figure 4), and each command involved manipulating components within the slide or view to achieve a particular effect – for example, zooming in or flipping items on the slide. The 'next slide' command initiated the next set of 8 commands on a new slide identical to the preceding one. Timing for each command selection consisted of the elapsed time since the previous correct command selection. All user actions were logged, and the whole experiment took participants 15 minutes on average. To gain insight into the behavior of our participants, we collected post-experiment comments using a short questionnaire. We had considered using a think-aloud protocol to achieve this, but since think-aloud protocols achieve the same goal as skillometers (i.e., promoting reflection during interaction) [17], it would have likely confounded our experimental objectives.

Design and Hypotheses

The experiment was designed as a 2×10 mixed-design ANOVA, with between-subjects factor *Interface* {*skillometer, control*} and within-subjects factor *Repetition* {*1..10*}. *Interface* was a between-subjects factor to minimize unintended learning effects. The two key dependent variables were proportion of hotkey use and task time (the time between command selections). Our primary hypotheses concerning skillometers were as follows:

H1: Proportion of hotkey use will be higher for *skillometer* than for *control*.
H2: Task time will be lower for *skillometer* than for *control*.
H3: There will be an *Interface* × *Repetition* interaction, showing more rapid hotkey adoption with *skillometer*.

Additionally, we wanted to ensure that the skillometer was addressing an appropriate target domain of user performance, leading to **H4:** hotkey selections will be faster than menu and toolbar selections.

Results

Hotkey Use

Figure 5-left shows the proportion of command selections completed using hotkeys for the control (left) and HotkeySkillometer (right) conditions across the ten repetitions. With HotkeySkillometer the mean rate of hotkey use across all repetitions was 50% compared to 28% for the control condition, giving a significant main effect of *Interface* ($F_{1,22} = 4.52$, $p < 0.05$) and supporting **H1**. There was

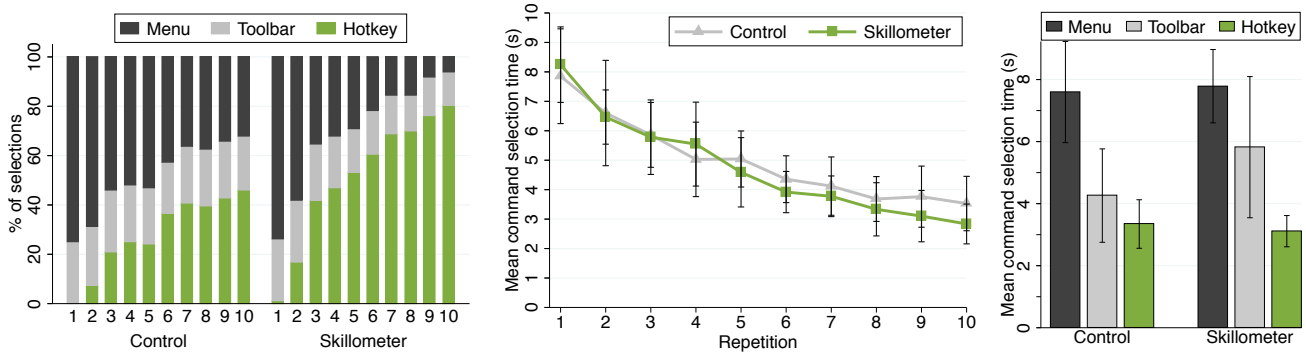


Figure 5. Percentage of selections per modality per repetition in both Control and Skillometer conditions (left). Mean command selection time across all repetitions (center). Mean command selection times across modalities and conditions (right).

also a significant interaction effect of *Interface*×*Repetition* ($F_{9,198} = 3.13$, $p < 0.01$), with participants making an earlier and more substantial switch to hotkeys when using the skillometer, with 80% of commands completed using hotkeys by the 10th repetition, compared to 42% with the control.

Command Selection Time

Figure 5-center summarizes the results for command selection time, showing that selections were initially marginally faster with the control but marginally faster in the skillometer condition across the later repetitions. Given a suggested cross-over effect, it is relatively unsurprising that there is no significant main effect for *Interface* ($F_{1,22} = 0.27$, $p = 0.6$). However, there is no significant *Interface*×*Repetition* interaction ($F_{9,198} = 0.55$, $p = 0.8$). We therefore reject **H2** (with caveats described in the Discussion).

Although there is no significant evidence that the skillometer reduced command selection time, analysis of variance for *Modality* (hotkey, menu, and ribbon) confirms that hotkey selections were much faster than toolbar or menu selections, with mean selection times of 3.3, 5.2 and 7.7 s respectively ($p < 0.001$). We therefore accept **H4**.

Figure 5-right, which shows the mean selection time for each modality in the control and skillometer condition, offers an explanation for the apparent inconsistency between the skillometer resulting in higher use of fast hotkeys, yet *not* resulting in a significant overall performance improvement. Specifically, it shows that toolbar selections were slower with the skillometer present (5.8 s compared to 4.2 s with the control).

One explanation for slow toolbar selections when using the skillometer is that users engaged in additional activities (mental and/or perceptual) in order to memorize and learn the hotkeys associated with commands, and that these activities consumed additional task time. Additionally, Figure 5-center suggests that skillometer users were still improving their performance and Figure 5-left further suggests that they were still increasing in the proportion of hotkeys used. Consequently, we suspect that participants' performance in the skillometer condition would have continued to improve as more hotkeys were learned and routinely executed. However, further empirical work is necessary.

DISCUSSION

To briefly summarise, skillometers are intended to be a widely applicable interface technique that motivates and assists users to increase their level of performance with an interface. They do so by encouraging users to reflect about their level of performance and by assisting them in understanding the performance benefits attainable by altering aspects of their interaction. We described four different domains of performance that skillometers can aim to improve (intramodal performance, intermodal performance, vocabulary extension and user strategies), and we identified four interface goals that any skillometer should seek to achieve (visualize the possibility to improve, provide details on how to improve, motivate the user to improve, and minimize disruption). We then described the design, implementation and evaluation of one specific skillometer addressing intermodal performance improvement – from point and click modality with menus and toolbars to hotkeys. Results of a lab study showed that users quickly learned and used more hotkeys with a commercial software product when presented with an accompanying skillometer.

While the experimental results are encouraging, they are preliminary and cover only a small subset of the potential domains into which skillometers could be deployed. In the following sections we discuss design refinements, explanations for our experimental results, and issues of generalisation.

Understanding and Refining HotkeySkillometer

In designing and implementing HotkeySkillometer, we followed the four interface design goals described earlier in the paper. Although the experimental results showed that the overall design successfully achieved increased hotkey use, it is difficult to know the relative contribution of each of the design goals in that success. This is an important question because the different interface design goals tend to pull the designer in different directions – for example, 'goal 4: minimize disruption' leads to subtle and transient visual effects that may go unnoticed, while goals 1-3 all encourage visual effects that are readily noticeable. Similarly, we are currently unable to determine whether each of goals 1-3 are crucial to the overall success of the system, or whether any one on its own would have had the same effect. For example, it is possible that the display of the hotkey bindings on the bar chart

(motivated by 'goal 2: provide specific details') accounts for all of the increase in hotkey adoption because it made hotkeys more salient in the display.

We attempted to mitigate some of these concerns in our experimental method by displaying only the last six commands within the skillometer (and therefore the previous use of any particular hotkey was not inside the view when next needed). We consider it unlikely that our results are attributable solely to the increased saliency of hotkey bindings, since previous research has failed to find benefits for such techniques [20]. We also attempted to understand the relative importance of different skillometer components by asking for subjective feedback on each, but this feedback yielded few strong insights. In general, participants viewed the history chart and overlaid hotkeys as being most useful, and the performance meter and motivational text least useful, with two stating that they did not look at the motivational text at all. We also noticed some participants tilting their head to read the text, suggesting that a horizontal chart would have been easier to read.

To elicit further feedback on the design we gave a real-world implementation of HotkeySkillometer to eight people to use on their personal computers for a week. While the lack of global toolbar support (see the Implementation section) reduced the usefulness of the tool, we received several comments on the design. One participant said he wanted to "measure improvement or long term performance", suggesting that the performance meter might have a configurable time component. Several participants suggested that the skillometer should not show hotkey selections, because it increases distraction when the user already knows the hotkey (counter to interface goal 4) and because it consumes space in the skillometer that might otherwise be dedicated to unknown hotkeys. However, two of the participants in the lab study would have disagreed with this recommendation, as they stated that hotkey feedback in the skillometer was useful for confirming which command was executed if they suspected an error was made. A related issue was raised regarding the inclusion of repeated command selections in the skillometer – a richer vocabulary of available hotkeys would be displayed if only the most recent repetition was displayed.

A final variation to the design of HotkeySkillometer that we are keen to pursue would blend the *feedback* methods that it currently employs with *feedforward* display techniques. HotkeySkillometer currently 'pushes' data to the user in response to their command selections. However there are rich interaction opportunities in supporting users who wish to explicitly 'pull' assistance prior to completing command execution. For example, ExposeHotkey [28] and OctoPocus [9] demonstrate improved learning of expert interface techniques by providing feedforward assistive information in response to partially complete interactions. A skillometer might use equivalent feedforward methods, such as displaying a list of frequent commands that have not been previously selected by hotkeys in response to the user pressing the command key.

Improving User Performance with Skillometers

An important issue for skillometers in general is whether they do in fact help users to improve overall performance – and in particular, why did this not occur in our initial evaluation? There are two main points for discussion in this issue.

First, the difference in completion time for toolbar selections between the skillometer and control groups may have arisen because groups were attempting to learn both the hotkeys and the toolbar locations at the same time – and in the skillometer condition, participants may have allocated more of their cognitive resources to learning the hotkeys, leading to less effective learning of the toolbar. There are often costs associated with learning new interaction modalities [35] – and it is worth noting that the skillometer group did not suffer any 'performance dip' as they switched to the new mode. More research is needed, however, to determine why the skillometer group continued to use the toolbar (or menu) in some instances – there may have been other types of 'interaction inertia' that future skillometer design could attempt to counteract.

Second, it is clear that hotkeys are substantially faster than the other two interaction modalities, so the observed slowdown in toolbar use for the skillometer group would likely be overcome by the hotkey advantage over a longer term than our 15-minute study (a trend that is also suggested by the divergence of the lines in Figure 5-center).

Challenges in Designing Skillometers for Other Domains

Although we provided design suggestions for each target domain earlier in the paper, it is clear that substantial challenges arise in designing and implementing skillometers for domains other than intermodal improvement, particularly for strategic deployments.

However, even in the complex domain of strategic improvement, there are realistic opportunities for successful deployment. In particular, computing systems are increasingly capable of capturing and usefully processing large pools of meta-information about users' interaction, both at individual and societal levels. These data repositories can be put to use to predict the user's intention and needs, and the predictions facilitate new forms of interaction to assist users in completing their tasks. For example, Fitchett and Cockburn [19] introduced the 'AccessRank' algorithm and demonstrated that it outperformed a variety of previous algorithms for predicting upcoming user actions across a wide range of user tasks. A skillometer could use such predictions to gain an understanding of the user's task, and suggest alternative methods for achieving it when alternatives are known. But again, interesting design questions emerge on whether the skillometer's role extends beyond identifying and suggesting better interactive alternatives (as we have described them), or whether they could additionally provide a shortcut method for achieving the predicted intention (which we see as beyond their scope).

Platform Requirements for Engineering Skillometers

Accessibility APIs remain the best solution for implementing systems like skillometers on a generic level. However, the main goal of these APIs is to be able to provide better access to interfaces for individuals with disabilities. These APIs are therefore usually limited to a shallow description of the UI components, and it is complicated (often impossible) to access deeper information such as the actions associated with

toolbar buttons. Ideally, the next generation of UI toolkits would provide an improved connection between UI components and their semantics.

CONCLUSIONS AND FUTURE WORK

Users have many opportunities to enhance their productivity in modern applications, one of them being to transition to expert modalities such as hotkeys. Yet users tend to stick to their familiar methods and strategies for interaction, rather than trying to improve. As a solution to this problem, this paper introduced the use of *skillometers*, a type of reflective interface that motivates and helps users to increase their level of performance. We provided a framework examining the domains of interaction that can be targeted for improvement, and discussed interface design goals for skillometers. We then described the design and evaluation of HotkeySkillometer, an exemplar skillometer that motivates and helps people to use more hotkeys. A preliminary lab study provided positive results, with experimental participants exposed to the skillometer learning more hotkeys than those in the control condition.

There are two main directions for future work. First, we will refine the design of HotkeySkillometer based on feedback from our study participants, and perform a longer-term, application-specific study with a complex application. Second, we will study skillometers in other settings and for other target domains, such as strategic improvement.

ACKNOWLEDGMENTS

This research was partially funded by Royal Society of New Zealand Marsden Grant 10-UOC-020.

REFERENCES

1. Apple accessibility api. https://developer.apple.com/library/mac/#documentation/Accessibility/Conceptual/AccessibilityMacOSX/OSXAXIntro/OSXAXintro.html.
2. Ribbon hero 2. http://www.ribbonhero.com/.
3. Adar, E., Tan, D., and Teevan, J. Benevolent deception in human computer interaction. In *CHI '13*, ACM (2013).
4. Aleven, V., Koedinger, K., Sinclair, H., and Snyder, J. Combatting shallow learning in a tutor for geometry problem solving. In *Intelligent Tutoring Systems*, Springer (1998).
5. Anderson, J., Corbett, A, Fincham, J., Hoffman, D., and Pelletier, R. General principles for an intelligent tutoring architecture. *Cognitive approaches to automated instruction* (Jan 1992).
6. Bailly, G., Pietrzak, T., Deber, J., and Wigdor, D. J. Métamorphe: augmenting hotkey usage with actuated keys. In *CHI '13*, ACM (2013).
7. Banovic, N., Chevalier, F., Grossman, T., and Fitzmaurice, G. Triggering triggers and burying barriers to customizing software. In *CHI '12*, ACM (2012).
8. Bateman, S., Teevan, J., and White, R. The search dashboard: how reflection and comparison impact search behavior. In *CHI '12*, ACM (2012).
9. Bau, O., and Mackay, W. Octopocus: a dynamic guide for learning gesture-based command sets. In *UIST '08*, ACM (2008).
10. Bhavnani, S., Peck, F., and Reif, F. Strategy-based instruction: Lessons learned in teaching the effective and efficient use of computer applications. *ACM Trans. Comput.-Hum. Interact. 15*, 1 (May 2008).
11. Bødker, S. Applying activity theory to video analysis: how to make sense of video data in human-computer interaction. In *Context and consciousness*, Massachusetts Institute of Technology (1995).
12. Carroll, J., and Rosson, M. Paradox of the active user. *Interfacing thought: Cognitive aspects of human-computer interaction* (1987).
13. Czerwinski, M., Horvitz, E., and Cutrell, E. Subjective duration assessment: An implicit probe for software usability. In *IHM-HCI '01*, vol. 2 (Jan 2001).
14. Deterding, S., Dixon, D., Khaled, R., and Nacke, L. From game design elements to gamefulness: defining "gamification". In *MindTrek '11*, ACM (2011).
15. Dong, T., Dontcheva, M., Joseph, D., Karahalios, K., Newman, M., and Ackerman, M. Discovery-based games for learning software. In *CHI '12*, ACM (2012).
16. Eagan, J., Beaudouin-Lafon, M., and Mackay, W. Cracking the cocoa nut: user interface programming at runtime. In *UIST '11*, ACM (2011).
17. Ericsson, K., and Simon, H. How to study thinking in everyday life: Contrasting think-aloud protocols with descriptions and explanations of thinking. *Mind, Culture, and Activity 5*, 3 (1998).
18. Evans, A., and Wobbrock, J. Taming wild behavior: the input observer for text entry and mouse pointing measures from everyday computer use. In *CHI '12*, ACM (2012).
19. Fitchett, S., and Cockburn, A. Accessrank: predicting what users will do next. In *CHI '12*, ACM (2012).
20. Grossman, T., Dragicevic, P., and Balakrishnan, R. Strategies for accelerating on-line learning of hotkeys. *CHI '07*, ACM (2007).
21. Kapp, K. *The Gamification of Learning and Instruction: Game-based Methods and Strategies for Training and Education*. Pfeiffer, 2012.
22. Krisler, B., and Alterman, R. Training towards mastery: overcoming the active user paradox. *NordiCHI '08*, ACM (2008).
23. Lane, D., Napier, H., Peres, S., and Sandor, A. 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 (2005).
24. Lesgold, A., Eggan, G., Katz, S., and Rao, G. Possibilities for assessment using computer-based apprenticeship environments. *Cognitive approaches to automated instruction* (Jan 1992).
25. Li, W., Grossman, T., and Fitzmaurice, G. Gamicad: a gamified tutorial system for first time autocad users. In *UIST '12*, ACM (2012).
26. Linton, F., and Schaefer, H.-P. Recommender systems for learning: building user and expert models through long-term observation of application use. *User Modeling and User-Adapted Interaction 10*, 2.
27. Mackay, W. Triggers and barriers to customizing software. In *CHI '91*, ACM (1991).
28. Malacria, S., Bailly, G., Harrison, J., Cockburn, A., and Gutwin, C. Promoting hotkey use through rehearsal with exposehk. In *CHI '13*, ACM (2013).
29. Matejka, J., Grossman, T., and Fitzmaurice, G. Patina: Dynamic heatmaps for visualizing application usage. In *CHI '13*, ACM (2013).
30. Matejka, J., Li, W., Grossman, T., and Fitzmaurice, G. Communitycommands: command recommendations for software applications. In *UIST '09*, ACM (2009).
31. Newell, A., and Rosenbloom, P. Mechanisms of skill acquisition and the law of practice. *Cognitive skills and their acquisition* (1981).
32. Odell, D., Davis, R., Smith, A., and Wright, P. Toolglasses, marking menus, and hotkeys: a comparison of one and two-handed command selection techniques. In *GI '04*, Canadian Human-Computer Communications Society (2004).
33. Peres, S., Tamborello, F., Fleetwood, M., Chung, P., and Paige-Smith, D. Keyboard shortcut usage: The roles of social factors and computer experience. *the Human Factors and Ergonomics Society Annual Meeting 48*, 5 (2004).
34. Salmoni, A., Schmidt, R., and Walter, C. Knowledge of results and motor learning: A review and critical reappraisal. *Psychological Bulletin 95*, 3.
35. Scarr, J., Cockburn, A., Gutwin, C., and Quinn, P. Dips and ceilings: understanding and supporting transitions to expertise in user interfaces. In *CHI '11*, ACM (2011).
36. Schmidt, R., and Lee, T. *Motor Control and Learning: A Behavioral Emphasis*. Human Kinetics, 2011.
37. Shroyer, R. Actual readers versus implied readers: role conflicts in office 97. *Technical Communication 47*, 2 (2000).
38. Simon, H. Satisficing. *The new Palgrave: a dictionary of economics 4* (1987).
39. Tak, S., Westendorp, P., and van Rooij, I. Satisficing and the use of keyboard shortcuts: Being good enough is enough? *Interacting with Computers* (2013).

Appendix B

Publication: Faster Command Selection on Tablets with FastTap

Faster Command Selection on Tablets with FastTap

Carl Gutwin[1], Andy Cockburn[2], Joey Scarr[2], Sylvain Malacria[2], and Scott Olson[1]

[1]Computer Science, University of Saskatchewan
Saskatoon, Saskatchewan, Canada

[2]Computer Science, University of Canterbury
Christchurch, New Zealand

carl.gutwin,scott.olson@usask.ca andy,joey@cosc.canterbury.ac.nz, sylvain@malacria.fr

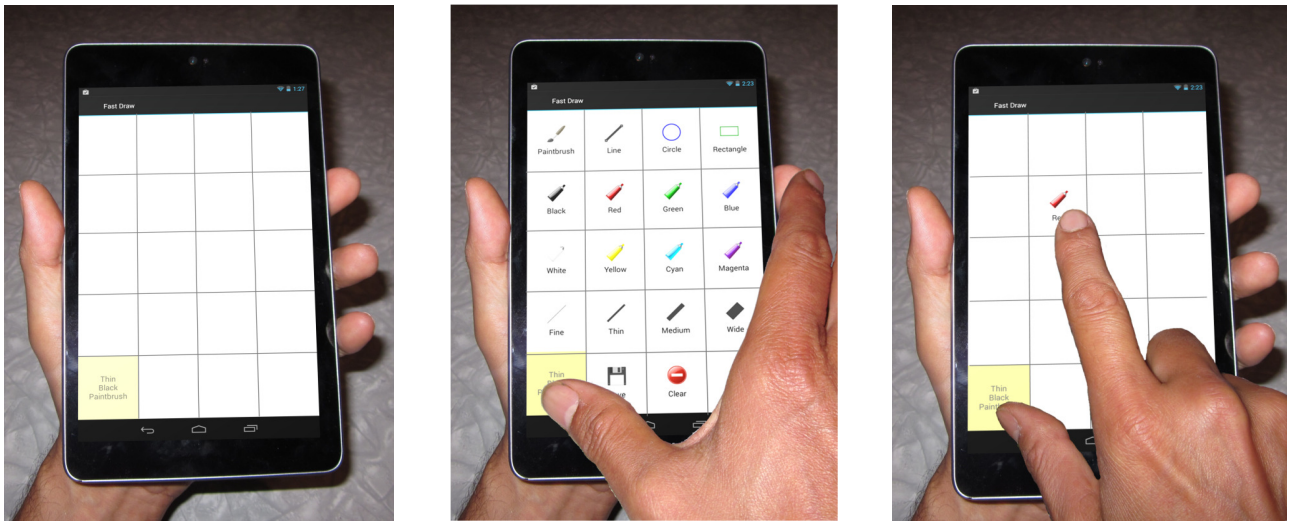


Figure 1. FastTap interface. Left: default state of the interface (gridlines enhanced). Center: FastTap grid overlay after touching the activation button. Right: FastTap selection by chording with the thumb and forefinger, without waiting for the overlay.

ABSTRACT

Touch-based tablet UIs provide few shortcut mechanisms for rapid command selection; as a result, command selection on tablets often requires slow traversal of menus. We developed a new selection technique for multi-touch tablets, called FastTap, that uses thumb-and-finger touches to show and choose from a spatially-stable grid-based overlay interface. FastTap allows novices to view and inspect the full interface, but once item locations are known, FastTap allows people to select commands with a single quick thumb-and-finger tap. The interface helps users develop expertise, since the motor actions carried out as a novice rehearse the expert behavior. A controlled study showed that FastTap was significantly faster (by 33% per selection overall) than marking menus, both for novices and experts, and without reduction in accuracy or subjective preference. Our work introduces a new and efficient selection mechanism that supports rapid command execution on touch tablets, for both novices and experts.

Author Keywords

Command selection; expertise; tablet UIs

CHI 2014, April 26–May 1, 2014, Toronto, Ontario, Canada.

ACM 978-1-4503-2473-1/14/04…$15.00.
http://dx.doi.org/10.1145/2556288.2557136

ACM Classification Keywords

H.5.2. Information interfaces (e.g., HCI): User Interfaces.

INTRODUCTION

Desktop interfaces often offer multiple ways to select the same command, and the different mechanisms can have very different performance characteristics. For example, selecting commands from menu hierarchies is slow when compared to keyboard shortcuts and toolbars, which provide access to commands with fewer actions. Having these types of shortcuts in an interface can substantially increase a user's efficiency over time, by allowing the user to learn quicker methods of invoking the commands they use most often.

On portable touch-based devices such as tablet computers, however, interface shortcuts are seldom available. The lack of a physical keyboard means that there are no keyboard shortcuts for quick selection, and limitations on screen real estate leave little or no room for always-visible components such as toolbars. Touch-based command interfaces, therefore, often take the form of tedious menu hierarchies, with no way of making a transition to an expert method of interaction. This greatly decreases the utility of touch devices for productivity tasks.

One widely-studied interface that supports expertise on touchscreens is the *marking menu*, a type of radial menu that allows visual inspection of menu items for novices, and rapid gestural interaction for expert users [12]. However,

the contact-move-lift actions for marking menu gestures may take longer to perform than a simple tap action. With multi-touch capabilities widely available in modern tablet computers, there are opportunities for interfaces that support rapid command execution for experts, as well as smooth transitions from novice to expert use [3, 16, 23].

In this paper, we present FastTap, a new rapid-access interaction technique that allows fast command selection on multi-touch devices for both novice and expert users. As shown in Figure 1, FastTap uses the entire screen to present a spatially-stable grid of commands (based on the recent CommandMap interface [20]). The command overlay is hidden by default, and is shown when the user holds their thumb on the grid activation button. The interface can then be inspected, and commands can be selected with a finger; when the user lifts their thumb, the grid disappears.

Novices use the interface by showing the grid and visually searching for the commands they need. As users become familiar with commands, however, they remember item locations in the grid, leading to expert behavior – experts can select a command with a single 'chorded' tap using the thumb and forefinger, without waiting for the grid to appear. Similar to marking menus, this design follows Kurtenbach's principle that *'guidance should be a physical rehearsal of the way an expert would issue a command'* [15] – in other words, since the novice and expert interaction methods require similar motor actions, users can develop spatial and muscle memory of the action required for each command through natural use.

To assess the performance of FastTap, we carried out a controlled study comparing selection time and errors with FastTap and marking menus. Our study showed that selection time was significantly and substantially faster with FastTap (mean 1.6 seconds per selection) than with marking menus (mean 2.4 seconds). In addition, FastTap was significantly faster at all levels of expertise with the interface, and provided an additional speed benefit when multiple commands were carried out in sequence. We found no differences in terms of errors, effort, or preference.

Our work makes three main contributions. First, we introduce a new interaction technique for tablets that was significantly faster than marking menus in an initial study. Second, we further demonstrate the power of spatial memory as an organizing principle for visual interfaces, and demonstrate how spatial memory can be exploited together with multi-touch input to produce rapid command selection interfaces. Third, we provide empirical results about user performance with FastTap and marking menus.

DESIGN GOALS AND RELATED WORK

As shown in Figure 1, the FastTap interface works by displaying a CommandMap [20] over the main workspace when the user places their thumb on an on-screen activation area. In this section, we discuss the design goals behind FastTap and frame them in the context of related work.

Design Goal 1: Enable rapid command execution

Modern touch-screen applications typically use hierarchical menus and dialogs, with a few commands accessible on the main display, and others requiring several pointing actions that slow their selection.

Gesture-based systems, such as marking menus [12, 14], are one alternative to hierarchical linear menus, with some implementations appearing in commercial products (e.g., Autodesk Sketchbook Pro). Marking menus allow practiced users to traverse a radial menu hierarchy in a single gesture, speeding up interaction. However, navigating the hierarchy can still be slow, and marking menus are inherently limited in terms of the number of items that can appear at each hierarchical level (see Design Goal 3 for extensions to marking menus that can reduce this problem).

Multi-touch technology has created new opportunities for designing efficient command selection interfaces that exploit the higher bandwidth available with multiple concurrent contacts. For example, Wu and Balakrishnan [24] describe multi-finger and whole-hand interaction techniques for tables, including a selection mechanism that posts a toolglass with the thumb, allowing selection with another finger. Similar techniques are used and studied in Wagner et al.'s BiTouch system [23], but with a focus on handheld tablets where the thumb of the non-dominant hand is used to post interface components.

Multitouch marking menus [16] and finger-count menus [3] both allow users to specify a menu category by changing the number of fingers used to touch the screen (thus reducing the number of levels that must be traversed). Other techniques parallelize the hierarchy traversal: for example, Banovic et al.'s multi-finger pie menu [4] allows users to post the menu with one finger and select an item with another; Kin et al.'s two-handed marking menus [10] allow users to draw the marks for different menu levels simultaneously, by using both thumbs. However, these higher-bandwidth techniques do not always improve performance, since a more-complex control action may take more time to retrieve and execute [10].

Techniques utilizing device features other than the touch-screen have also been implemented. Jerktilts [1] allow quick selection of one of eight commands by quickly tilting the mobile device. Bezel Tap [22] uses the accelerometer to detect taps on the edge of the device, allowing shortcuts to commands while the device is asleep.

One key characteristic that determines whether a command selection interface is efficient is the number of separate actions needed to navigate to an item. Reducing this number is a main design goal of Scarr et al.'s CommandMap [20] technique, which uses a full-screen overlay to display as many commands as possible at once. These commands can be selected with a single action, which is faster than the multiple navigational steps needed for hierarchical menus and ribbons. Scarr et al. also showed

that navigational errors (e.g., choosing the wrong menu) substantially increased the time needed for hierarchical organizations. In this work, we adapt the CommandMap's flat and spatially stable design to work with mobile devices.

Design Goal 2: Support a transition to expertise

One of the primary advantages of marking menus is the way in which they support a smooth and rapid transition to expert use [12]. After activating a marking menu, a novice user can wait for a short time to see a labeled radial menu appear, from which they make their selection with a touch gesture; an expert user can make the same gesture without waiting for the menu to appear. Since the motor actions for the novice and expert uses of the menu are identical, users learn the expert gestures through normal interaction.

This principle of *rehearsal* is extremely important to the development of user expertise. FastTap is therefore designed to support rehearsal during novice use. In a similar manner to marking menus, the command grid only appears on-screen after a delay; users with spatial or muscle memory of the interface can interact instantly without waiting for the visual display, using the exact physical action they used as novices; intermediate users suspecting the desired command location but unwilling to execute it without confirmation can also benefit from FastTap, by anticipating the location of the target, positioning their finger over it while waiting for the grid to appear, and selecting the command after visual confirmation.

Design Goal 3: Support a large number of commands

While marking menus are generally limited to eight or twelve commands per level [14], various extensions significantly increase this limit. Polygon menus [25] and flower menus [2] both allow more commands by increasing the types of gestures available. More recently, Roy et al. [19] developed Augmented Letters, a system whereby users draw the first letter of a command on the screen, then select from a radial menu of resulting candidate commands. OctoPocus [5] recognizes gestures by shape, and provides visual suggestions for the remaining gesture based on the initial movements. While these systems increase the number of commands that marking menus can support, they still rely on gesture-based interaction.

Rapid execution is our priority for FastTap, but we also intend that it will support a wide command vocabulary. In FastTap, the number of items at each level is limited only by the size of the screen; our prototype uses a 5x4 grid, with one cell being used as the FastTap activation button. However, this number can be increased through the use of different activation buttons, or command tabs, which can be arranged along the bottom of the screen. We consider these design possibilities further in the Discussion.

STUDY: SELECTION PERFORMANCE

The aim of the study was to assess the performance of FastTap for command selection on tablets. We compared FastTap to marking menus [15], which allow fast command selection for experts and support a smooth transition to expertise. We compared the two interfaces in a controlled experiment where participants selected a set of commands over several repeated blocks, allowing us to examine both novice and more expert selection behavior.

Experimental Conditions

Both FastTap and marking menus were implemented in a functional multi-touch drawing application (see Figure 2).

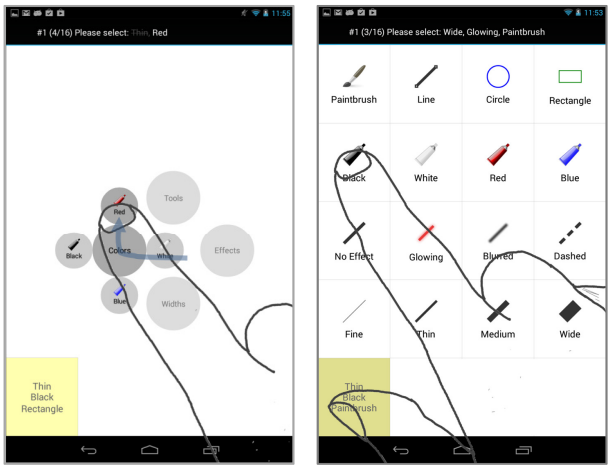


Figure 2. Study UIs. Left: marking menu (arrow shows gesture path). Right: FastTap. Cue appears at top of screen.

FastTap. As described above, FastTap provides modal access to a grid of command buttons. Selections are made by pressing a command button, either after invoking the grid display, or simultaneously with the invocation button (i.e., by chording). There is no difference in the selection mechanism for novice or expert use – experts who know the item locations simply tap the command before the interface is shown. After a chorded selection, feedback on the selected command is given by displaying the command icon for 500ms (Figure 1, right). The interface used in the experiment contained sixteen command buttons in a 4x4 grid , of which eight were used as study targets. The sixteen commands were organized into four rows that grouped similar commands together (see Figure 2).

Marking menus. We implemented a 16-item marking menu with a two-level hierarchy, adapted from Kurtenbach's previous work [15]. Once again, only eight of the 16 items were used as study targets. Upon invocation of the menu (described below), users move a finger towards one of four categories shown on screen (Shapes, Colors, Line Style, Line Width), and then to one of the items in that category (see Figure 2, left). The items in each category were the same as the row groups used in FastTap (Figure 2). We used a hierarchy because previous research indicates that selection errors increase significantly for menu levels containing more than eight items [15]. We chose commands for the study that fit well with the four categories, to reduce the chance that participants would select the wrong category. The categories are only needed when participants

are new to the menu, however; as users become familiar with item locations in the marking menu, they can shift to expert interaction simply by drawing the gesture that uniquely indicates an item (e.g., up-right).

Marking menus also need an invocation mechanism, and we used the same activation button as in the FastTap condition (see Figure 2). Users invoked the menu by touching the activation button with their thumb, then touching their finger on the screen to instantly post the menu. Moving the finger in one of the four directions then opens the corresponding submenu, and moving again selects one of four items. Some implementations of marking menus use a 'press timeout' invocation, but to avoid disadvantaging the menu condition, we chose a method that was equally fast to the invocation of FastTap.

Procedure

Participants completed a demographics questionnaire, and then performed a sequence of selections in a custom study system with both marking menus and FastTap. For each trial, a command stimulus (consisting of one, two, or three command names) was displayed at the top of the screen; the participant then selected the command(s) using the interface provided. Trials involved selecting a combination of one, two, or three individual tools and properties that could be used within the drawing application (e.g., 'Red', 'Red Line', 'Red Thin Line'). Trials were timed from the appearance of the stimulus until all targets were successfully selected. In the case of multiple-command targets, command names in the stimulus were crossed out as they were selected, and participants could select commands in any order. Participants were instructed to complete tasks as quickly as possible, and were told that errors could be corrected simply by selecting the correct item. Completion times included the time for correcting errors.

Of the sixteen commands in each interface, only eight were used as stimuli, in order to allow faster development of spatial memory and expertise for both interfaces. Two commands were used from each interface category (Shape, Color, Line Style, Line Width). Multiple-command targets were also composed from these eight commands.

For each interface, selection trials were organized into blocks of sixteen selections (eight one-command targets, four two-command targets, and four three-command targets). Participants first performed one practice block of sixteen trials (data discarded) to ensure that they could use the interfaces successfully. They then carried out 10 blocks of sixteen selections. Targets were presented in random order (sampling without replacement) for each block. Short rest breaks were given between blocks.

After each interface, participants filled out a NASA-TLX questionnaire [8]; at the end of the study, they also answered summary questions about their preferences.

Participants and Apparatus

Sixteen participants were recruited from a local university (8 female; mean age 26.2 years). The study was conducted on a Nexus 7 Android tablet (7-inch screen, 1280x800 resolution). The software was written in Java, and recorded all experimental data including selection times and errors.

Design and Hypotheses

The study used a 2×3×10 within-participants RM-ANOVA with factors *Interface* (FastTap, MarkingMenu), *NumberOfCommands* (1, 2, or 3 commands per trial), and *Block* (1-10). Dependent measures were command selection time, and errors per command selection. Interface was counterbalanced between participants. Hypotheses were:

- **H1.** Mean selection times for *FastTap* will be faster than for *MarkingMenu*.
- **H2.** *FastTap* will be faster than *MarkingMenu* both for novices and for experts.
- **H3.** *FastTap* will show an added speed benefit for two- and three-command targets.
- **H4.** There will be no evidence of a difference in error rates between the two interfaces.
- **H5.** There will be no evidence of a difference in perception of effort for the two interfaces.
- **H6.** Users will prefer *FastTap* over *MarkingMenu*.

Results

For significant ANOVA results, we report partial eta-squared (η^2) as a measure of effect size (where .01 should be considered small, .06 medium, and > .14 large [5]).

Selection Time per Command

We calculated the selection time for each command by dividing the total trial time by the number of commands in that trial. As shown in Figure 3 (rightmost bars), mean selection times were about 0.8 seconds faster per command with *FastTap* (1.60s, s.d. 0.52s) than with *MarkingMenu* (2.39s, s.d. 0.65s), giving a significant main effect of *Interface* ($F_{1,15}$=84.37, p<.0001, η^2=0.85). We therefore accept **H1** – FastTap selections were 33% faster overall than the marking menu.

Figure 3 also shows selection time by the number of commands per trial. ANOVA showed a significant main effect of *Number of commands* ($F_{2,30}$=3.7, p<.05, η^2=0.20), and also an interaction with *Interface* ($F_{2,30}$=6.69, p<0.005, η^2=0.31). As suggested by Figure 3, both effects can be attributed to *FastTap* permitting faster selections when commands are joined into groups of two or three. Post-hoc t-tests (Bonferroni corrected) show a significant difference between the two interfaces for each number of commands (all p<.0001). We therefore accept **H3**.

Learning Effects

As shown in Figure 4, selection times decreased across trial blocks; ANOVA showed a significant main effect of *Block* ($F_{9,135}$=17.49, p<.0001, η^2=0.54). There was no interaction with *Interface* ($F_{9,135}$=1.43, p=.183) or with *NumberOfCommands* ($F_{18,270}$=1.29, p=.194).

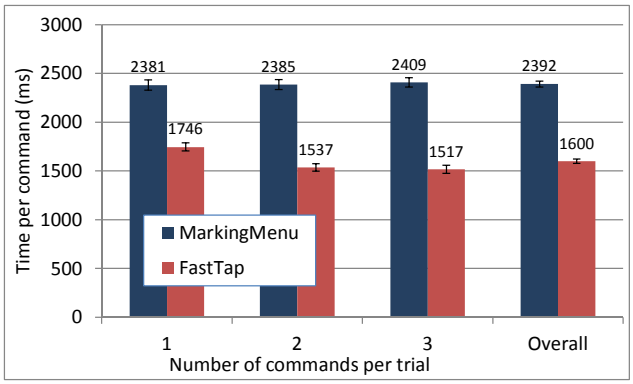


Figure 3. Mean selection times by Interface and Number.

We also analysed the performance difference between interfaces at each block. Bonferroni-corrected t-tests showed that *FastTap* was faster than *MarkingMenu* at all levels of expertise (all p<.001). Since the advantage of FastTap over marking menus is consistent across trial blocks, we accept **H2.**

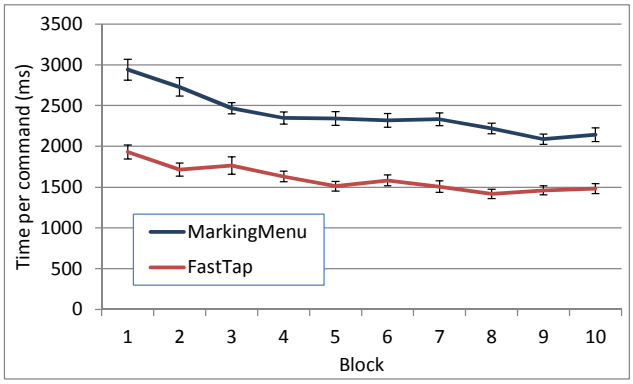


Figure 4. Mean trial times by Interface and Block.

Errors

As with selection time, we analysed errors per command, dividing the number of errors in a trial by the number of commands in that trial. Errors were counted as any incorrect selection (note that multiple-command trials could be carried out in any order). ANOVA showed no effect of *Interface* on errors, with *FastTap* at 0.10 errors/command, s.d. 0.13, and *MarkingMenu* at 0.08 errors/command, s.d. 0.11 ($F_{1,15}$=2.96, p=.11). We therefore accept **H4** (errors are considered further in the discussion section).

There was also no effect of *Block* ($F_{9,135}$=0.41, p=.93) or *NumberOfCommands* ($F_{2,30}$=0.51, p=.61) on errors per command; there were no interactions between any factors.

Subjective Responses: Effort and Preferences

User response was positive to both interfaces, but with no strong differences in NASA-TLX scores (compared using the Friedman test; see Table 1). No significant differences were found on any effort question, and the mean scores were close in all cases; therefore, we accept **H5**.

	FastTap	Marking Menu	χ_r^2	*p*
Mental demand	5.3 (2.2)	5.1 (2.0)	0.25	.62
Physical demand	3.8 (2.0)	4.4 (1.8)	0.25	.62
Temporal demand	4.6 (2.0)	4.7 (1.7)	0.0	1.0
Success	6.6 (1.3)	6.5 (1.4)	0.0	1.0
Hard work	5.3 (1.8)	5.5 (1.5)	0.25	.62
Frustration	3.6 (1.8)	3.5 (1.6)	0.0	1.0

Table 1. Mean (s.d.) NASA-TLX scores (0-9 scale, low to high).

We also asked participants which interface they preferred in terms of several qualities (see Table 2). As with the effort scores, counts were very close with no quality showing a significant difference. When asked about overall preference, seven participants preferred FastTap, eight preferred marking menus, and one had no preference. We therefore cannot accept **H6**.

	FastTap	Equal	Marking Menu	χ_r^2	*p*
Speed	8	2	6	0.071	.79
Accuracy	6	1	9	0.26	.61
Memorization	6	2	8	0.071	.79
Expert mode	5	3	8	0.31	.58
Comfort	8	1	7	0.0	1.0
Overall	7	1	8	0.0	1.0

Table 2. Counts of participant preferences.

Participant Comments

Like the preference results, participant comments were about evenly divided between the two selection techniques, and participants often mentioned the characteristic features of the designs when explaining their preferences. For example, participants made several comments about how spatial stability and quick activation helped the speed of FastTap: one person commented on *"the simple and stable location of each icon"*; another said *"the grid is faster because it only requires a two finger tap, with one finger always in the same place. The menu, however, demands specific movements (swipes) which expend a few milliseconds more time."*

It was also clear that some participants liked the semantic categories of the Marking Menu, and others liked FastTap's access to all commands at once. In favour of marking menus, participant comments included *"the menu compartmentalized the options much better"*; *"the menu required less memory"*; and *"visually a grid with all available options is not as easily navigable as a menu with divisible submenus."* In favour of FastTap, comments included: *"the options are all available at one time which makes it easier to pick, and memorization also helps a lot in the grid"*; *"it is easier to see the items that you want to select"*; and *"there is only one level to be memorized, where [the marking menu] has two levels."*

Some of the participants were initially concerned with the number of commands in FastTap, but ultimately preferred it: *"after a while, the grid became easier to remember"*; "*I was able to catch on much quicker [with FastTap] and be able to visualize where everything was."*

Finally, one participant also mentioned that the sliding motions needed for the marking menu could present problems on a touch surface, and that FastTap's tapping

action did not have this problem: *"sliding over a surface can be impeded by many factors like moisture [on the] hand and friction over the surface. […] I feel the tapping action in the grid is more comfortable."*

ADDITIONAL INFORMAL USAGE OBSERVATIONS

To gather additional information about whether FastTap would work outside of a study setting, we developed a simple multi-touch painting application with a FastTap interface and showed the system to five people in an informal test. We were interested in whether people would hold the device in a way that allowed correct FastTap operation (building on the work of Wagner et al. [23]), whether people would be able to use the interface with a minimum of instruction, and whether people would take initial steps toward expert selection behavior.

The five participants used the interface in a variety of settings (an office, an undergraduate lab, and three homes), and we asked them to play with the interface and to recreate a printed picture three times (to simulate increasing familiarity with a set of commands).

Holding the device. Wagner et al. [23] looked at the ways that people held an iPad, in order to determine whether the non-dominant hand could be used for bimanual input (e.g., to show toolbars or other interface elements). They observed several common postures involving the non-dominant palm or forearm. We also saw several different handholds in our informal sessions, some of which were seen by Wagner – for example, the tablet held in the left palm, with the FastTap activation button pressed by the non-dominant thumb. However, several other people held the device by gripping it between their left thumb and fingers – which was not possible in Wagner's study due to the size of the iPad, but which was possible with our seven-inch Nexus tablet. This grip-based hold allowed operation of FastTap with the thumb and fingers of the dominant hand, which matched our intention in designing the method.

Novice and more expert behaviour. In our informal sessions, people found it easy to use FastTap – their hands fit the interface reasonably well, and people did not have any difficulty finding the commands that they wanted. On a few occasions, however, people inadvertently drew rather than selecting. In the short time of these sessions, we saw only a few uses of expert chording behaviour; however, it was clear that all users were gaining spatial memory of frequently-used commands. For example, people moved their selection finger towards or over the target even before displaying the grid, showing that they had an idea about where the desired command was located.

These preliminary observations are limited, but provide initial evidence that FastTap can work in natural use.

DISCUSSION

Our investigation of FastTap provides four main results:

- Selection with FastTap was significantly faster overall than with marking menus (0.8s per selection, more than 33%), and it was faster throughout the learning curve;
- Making multiple selections allowed a speedup of about 200ms in selection time for FastTap, which did not occur with marking menus.
- There was no significant difference in errors between FastTap and marking menus, although FastTap had a slightly higher error rate (0.10 errors/selection compared to 0.08 for Marking Menu);
- Participants were evenly divided between the two interfaces for perceived effort and overall preference; some participants particularly liked each interface;
- In limited usage observations, we saw that people were able to hold the device appropriately for FastTap, were able to find commands easily, and appeared to quickly build memory of command locations.

The following sections provide explanations for these findings, address design issues for real-world use of FastTap, and discuss its generalizability.

Explanation and Interpretation of Results

Why did FastTap work well for both novices and experts?

Our experiment showed that FastTap improved (by 33%) on a well-known technique (marking menus) for selecting commands on tablets; in addition, FastTap was faster at all stages of learning.

The performance advantage of FastTap can be explained through an analysis of the steps required for command selection, in both novice and expert cases. For novices, there are three steps required: activating the command mode, searching for the desired command, and executing a selection action. Activating the command mode was the same in both interfaces, but the interfaces differed in the second and third steps. Searching for a command in FastTap involves only visual search over all of the concurrently displayed items, whereas search in marking menus involves first choosing and selecting a semantic category (Colours, Tools, etc.) and then searching in a much smaller set of items. Although some participants found the semantic categories helpful, previous research has shown that the decision and selection time costs of traversing the hierarchy can exceed that of a broader visual search [20]. The visual search needed for FastTap could take longer initially, since there are more items to inspect, but the rapid development of spatial memory means that novice users will quickly make a transition from full visual search to remembering where items are located. The third step – execution of the selection action – is faster in FastTap, since a tapping action can be carried out more quickly than the sliding motion of a marking menu.

For experts, selection requires only two steps: retrieval of the command action (either a location or a gesture) from memory, and execution of that action. The performance advantage for expert use of FastTap most likely arises from

the speed of execution, since the memory-retrieval step is similar for both techniques. Execution of a thumb-and-finger tapping is a faster action than a drawn gesture, leading to a performance advantage for FastTap.

Why were multiple selections faster than single selections?
Multiple commands selected with FastTap were each about 200ms faster than single commands. We believe that this speed-up is due to people's ability to visualize multiple command locations within FastTap's grid interface, and optimize their movements for multiple selections. For example, we observed participants re-ordering the selections to reduce finger movement (e.g., ordering the selections by grid row allowed people to move their finger in a straighter line). With the marking menu, there is less opportunity for optimization (e.g., re-ordering the commands does not reduce overall movement).

Does FastTap lead to more errors?
Error rates were high overall in both techniques (10% and 8% for FastTap and marking menus). This high error rate is probably an artifact of our experimental protocol, which explicitly instructed participants to select items as quickly as possible, while noting that errors could be corrected afterwards. Although FastTap had a slightly higher mean error rate (10%) than marking menus (8%), the difference was not statistically significant (p=.11). However, if we assume that effect of higher errors with FastTap is actually a reliable one, we see three candidate explanations. First, the quick execution of a selection action in FastTap may have encouraged participants to view errors as amenable to rapid correction, thereby encouraging users towards a 'guess and correct' mode of operation. Second, participants may have found the post-selection visual feedback in FastTap more clearly communicative of the selected item than the marking menu, again encouraging faster but more error prone selections. Third, it is possible that people's memory of an item's spatial location was imperfect, and so participants may have experienced 'near misses' more often than with marking menus. Further work is needed to determine whether the difference in error rates is reliable, and to properly explain its cause if it is.

FastTap and Marking Menus with Larger Item Sets

Our study tested FastTap and marking menus with a small command set (sixteen items), and it is worth considering how the two approaches would compare in the case of a larger interface. The main performance differences between FastTap and marking menus are in initial visual search (novice behavior) and the number of actions needed for a learned item (expert behavior). These differences arise due to the properties of FastTap's single-level presentation, compared with marking menu's hierarchical organization.

With a larger command set, it seems likely that FastTap will require more visual search than marking menus – provided the hierarchical organization of the marking menu items is clear (which can be a non-trivial design problem [20]). However, several prior studies suggest that the novice period of use is relatively short (e.g., [20, 21]) and users will soon be interacting with commands that they are familiar with. In this case, FastTap selection time will be similar to that observed in our study – since only a single invocation and selection action is needed, regardless of the size of the command set. Marking menus, however, must create hierarchies because they use less space, and so experts must continue to execute multiple navigational steps (even if these are executed as marks).

Recent work on multi-touch marking menus [3, 10, 16] suggests ways that these navigation interactions can be speeded up – e.g., by encoding top-level menu categories with different finger postures [16], or by parallelizing the execution of navigational marks [10]. Although we believe that FastTap will continue to compare well against other selection techniques as command sets increase in size, further work is needed to explore the potential differences between the spatial-memory approach of FastTap and the gesture-memory approach of marking menus.

Design issues for FastTap

In the following sections we consider issues likely to arise if deploying FastTap in real-world tablet interfaces.

How many commands can the interface accommodate?
Our drawing interface contained 19 commands in a 5×4 grid, but this is not the limit for the FastTap approach. In general, the number of commands in the interface is limited by the size of the device, the minimum desired size of the targets, and the size of the user's hand (to facilitate chording). Using the average width of an adult index finger (16-20mm [7]) as a guideline, it would be possible to have a grid of up to 40 buttons in an 8×5 layout on a 7-inch tablet. If we consider Parhi et al.'s recommendation that touch targets be no smaller than 9.6mm [17], a maximum density of 150 items in a 15×10 arrangement is possible. However, this size guideline is debatable – larger button sizes have been shown to provide higher success and satisfaction rates [17, 18], yet much smaller targets can also be used, as demonstrated by smartphone virtual keypads (which can be operated substantially eyes-free with sufficient practice). There are interesting research questions around users' ability to form spatial and motor memories for varying numbers of targets at different target densities.

Importantly, however, the possibilities for adapting FastTap to high functionality applications, or to small displays, is not necessarily limited to the 'all commands at once' designs examined in this paper. The capacity of the interface to display candidate targets can be multiplied through the use of multiple trigger buttons, or 'tabs', which organize the commands into multiple categories – each trigger button would then show a separate set of commands (see mockup in Figure 5). As discussed under Design Goal #3, this technique requires the allocation of additional trigger-button regions in the bottom row of the grid. Further research is required to determine how well people can remember these two-finger combinations.

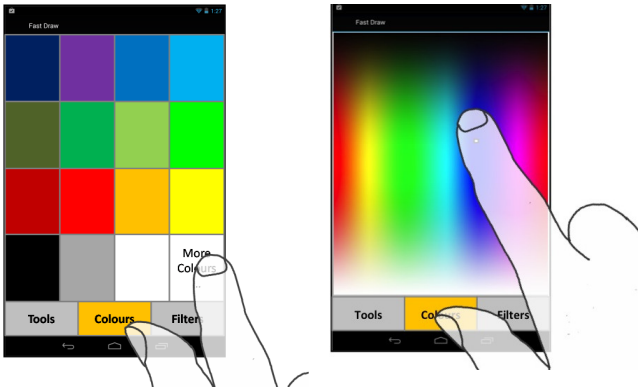


Figure 5. Left: FastTap UI with three tabs. Right: A color picker converted to work with FastTap.

Finally, it is also possible that if the grid interface cannot accommodate all of the application's commands, it could still function as a 'hotbox' [13] for frequently-used items.

FastTap with different size factors

Our experiments were conducted on a device equipped with a 7-inch touchscreen. We used this relatively small size factor because tablets with screens smaller than 8 inches are the main proportion of the current tablet market, and this proportion is forecast to increase by 2017 [9]. We conducted pilot studies on several smaller touch-screen devices (e.g., Nexus 4, Galaxy S3), mostly testing the drawing system. The systems worked well, although the command buttons were smaller overall, which may have accuracy implications for expert users.

Porting FastTap to tablets with *larger* screens also raises interesting questions, especially whether the designer should use the full screen to fit more commands. If not, the FastTap interface can be limited to a comfortable size for the user's hand. Using the full screen, however, would permit more commands to be displayed, but it would also create problems for single-handed operation due to hand-span limitations. A different triggering mechanism might be used for commands that are further away than a hand span – such as the bimanual thumb technique used in Wagner's BiTouch system [23].

Issues regarding device orientation

Mobile devices such as tablet computers can be used in different orientations, which changes the aspect ratio of the screen. There are three possibilities to accommodate orientation changes. First, the grid could maintain its overall aspect ratio and scale to fit the smaller dimension of the new orientation, requiring that users adapt to a different-scale interface. Second, the grid could change its aspect ratio to fill the new orientation, requiring that users adapt to a stretched version of the grid. As shown by Scarr et al. [20], these two transformations would cause only a minor disruption to spatial selections. Finally, the grid could maintain its size regardless of the orientation, fitting the most constraining orientation.

Biomechanical constraints

Some of our participants cited difficulties with the hand positions required to select commands with FastTap. Specifically, participants noted that targets were more difficult to acquire when they were close to or far away from the trigger button, or when they were at the lower-right side of the grid. People with long fingernails had particular difficulty maintaining registered contact with the display when selecting targets near to the trigger button, (the natural posture for such selections involves the fingers being nearly perpendicular to the display, causing nails to lift fingers off the touch screen).

In all cases, however, problems with hand positions were addressed in two ways. First, users could re-orient the device with the holding hand in order to present a better presentation of the trigger and target buttons. For example, to more easily tap a target in the lower right part of the grid, the user could turn the device slightly anticlockwise with their holding hand, so that the target button (finger) was above and to the right of the trigger button (thumb). Second, users did not have to use the thumb and forefinger to make selections, and could use a combination that simplified the particular selection. For example, many users used their first and second fingers for the bottom row of commands instead of their thumb and first finger.

It would also be possible to change from a grid-based command organization to one that better matches the ergonomics of the hand. However, it is not clear whether a non-rectilinear grid would make it more difficult to acquire targets in expert mode (i.e., with no visual feedback).

FastTap and advanced interfaces

Modern UIs generally include widgets more advanced than push-buttons. For example, sliders or color spectrum pickers can be used to provide a finer degree of control over application parameters. These widgets can easily be converted to work with FastTap, as shown in Figure 5. FastTap provides quick access to these widgets and allows users to perform rapid and coarse slider positioning that can instantly be adjusted without having to switch modes.

Modal interface and occlusion of the document

All in-place interfaces (including marking menus) occlude parts of the document, but the overlay presentation of FastTap's interface is different in that it completely fills the screen. In most cases, this is unlikely to cause a problem for the user, for three reasons: first, the grid is partially transparent and so the main visual elements of the document can be seen through the FastTap overlay; second, the overlay does not appear at all in expert mode, so there is no occlusion problem for experienced users; third, it is easy for the user to control the presence of the overlay (by lifting their thumb from the activation button), allowing rapid switching back and forth from the document to the UI.

One case where the overlay presentation requires a design solution is the situation where the user has selected an

object on the screen, and wants to see previews of a command's effects on that object (e.g., what it looks like in different colours). Although it is possible to simply select different colours using the standard FastTap process (expert mode makes this fast and easy), other designs could reduce the problem – such as placing a floating copy of the selected object above the grid, or fading out the overlay whenever the user's finger stops moving across the grid.

Last, the visual presence of the activation button is another potential cause of occlusion. This button is also partially transparent, and could easily increase in transparency level as the user becomes more experienced (although this does not happen in our current prototype).

Multiple simultaneous selection

Normal expert selection in FastTap involves two digits (normally the thumb and one finger; but sometimes two different fingers). This combination selects one command; however, there is no reason why additional commands cannot be selected simultaneously in the same chorded tap. For example, the user could select "black paintbrush" with the three-finger chord shown in Figure 6.

These kinds of selections already work in our prototype application. Added-finger selections work well only for certain combinations (because of constraints on positioning the fingers); therefore, if these are to be used in interface design, further work must determine which fingers combinations are possible, and which command combinations are desirable.

Error rates in memory-based UIs

Above we discussed error rates with FastTap and marking menus. However, a more general question concerns the errors that will always occur with a memory-based means for interaction. Errors are inherent in any memory-based technique, particularly in the transition from novice to expert use, since users must at some point depend on their memory rather than on visual feedback. This issue affects marking menus, keyboard shortcuts, and FastTap.

In general, there are several issues that must be considered when designing for the possibility of errors: the destructiveness of a command, the ease with which the error can be identified, the time that the command takes to execute, and the time required to correct the error. For non-destructive and instantaneous commands such as the property-setting commands of a drawing program (colour, line style, etc.), selection errors are a minor problem. If feedback on the selection is given (as it is in our implementation, even in expert mode), users are likely to realize their error and simply correct it.

For commands that take a long time to construct or execute (e.g., a complicated image filter), commands should be easily interruptible to avoid problems of erroneous selection. For destructive commands (e.g., deletions), systems may need a confirmation step or a well-designed 'undo' facility. It is possible that memory-based techniques like FastTap or marking menus could be reserved for lighter-weight commands, with destructive or time-consuming commands accessed through a different mechanism (e.g., a traditional application menu).

Figure 6. Left: transparent grid with drawing behind. Right: three-finger chord that selects two commands.

Single-handed use

Although we developed FastTap on a tablet that would be difficult or impossible to use with one hand, several people have noted that phone-sized mobile devices are often used in a one-handed fashion. The ideas underlying FastTap can be used in a one-handed context, providing some or all of the benefits of the interface. The spatial-memory advantages of FastTap could be achieved even without multi-touch chorded taps; a two-tap version of the interface would bring up the interface with one tap, and then allow selection with a second tap. For most single-handed operation, the user touches the screen with their right thumb, which means that the activation button would have to be moved to the right side of the screen.

It would also be possible to place an activation button somewhere else on the device to accommodate chording with one-handed use (e.g., on the back of the device, or on the bezel under one of the other fingers, although these would require hardware changes). Similarly, the activation functionality that currently uses a multi-touch selection could be changed to another modality – for example, double-tapping on a grid location could be a method for specifying both activation (the double tap) and selection (the location) with a single finger.

CONCLUSIONS AND FUTURE WORK

Although multi-touch tablets are now common, and are starting to be used for productivity work, there are few techniques for these devices to support rapid command selection. In this paper, we presented a new selection technique for multi-touch tablets called FastTap that uses thumb-and-finger touches to show and choose from a grid-based overlay interface. FastTap allows novices to view and inspect the full interface, but once item locations are known, FastTap also allows people to select commands

with a single quick thumb-and-finger tap. The interface helps users move toward expert use, since the motor actions carried out in novice mode rehearse the expert behavior. A controlled study with 16 participants showed that FastTap was significantly faster (0.8 sec/selection, 33%) than marking menus, both for novices and experts, and without reduction in accuracy or subjective preference.

Our research thus far with FastTap gives us strong initial results, and future work will continue in three directions. First, we will continue development of the drawing application and release a fully-functional version of the app, in order to gather real-world usage and performance data from a wide audience. Second, we will develop new prototypes that explore some of the design issues described above, including tabbed command sets, complex interface widgets, and interface scaling for larger devices. Third, we will develop FastTap prototypes for other applications that could benefit from fast access to commands and shortcuts (e.g., contacts and response options in a mail client, or favourites and page actions in a web browser). Our experience with FastTap suggests that the underlying ideas of spatial-memory-based expertise and quick tap-based access can be successfully and broadly applied across several application domains.

ACKNOWLEDGMENTS

This work was supported by the NSERC SurfNet Research Network and the GRAND NCE, and by the Royal Society of New Zealand Marsden Grant 10-UOC-020.

REFERENCES

1. Baglioni, M., Lecolinet, E. and Guiard, Y. JerkTilts: using accelerometers for eight-choice selection on mobile devices. *Proc. ICMI 2011*, 121-128.
2. Bailly, G., Lecolinet, E. and Nigay, L. Flower menus: a new type of marking menu with large menu breadth, within groups and efficient expert mode memorization. *Proc. AVI 2008*, 15-22.
3. Bailly, G., Müller, J. and Lecolinet, E. Design and Evaluation of Finger-Count Interaction: Combining Multitouch gestures and Menus. *IJHCS* 70(12):673-689.
4. Banovic, N., Li, F., Dearman, D., Yatani, K. and Truong, K. Design of unimanual multi-finger pie menu interaction. *Proc. ITS 2011*, 120-129.
5. Bau, O., and Mackay, W., OctoPocus: a dynamic guide for learning gesture-based command sets. *Proc. UIST 2008*, 37-46.
6. Cohen, J. Eta-squared and partial eta-squared in communication science. *Human Communication Research* 28(56):473-490.
7. Dandekar, K., Raju, B. and Srinivasan, M. 3-D finite-element models of human and monkey fingertips to investigate the mechanics of tactile sense. *Journal of Biomechanical Engineering*, 125, 5, 2003, 682-691.
8. Hart, S. and Staveland, L. Development of NASA-TLX (Task Load Index): Results of Empirical and Theoretical Research. in Hancock, P. and Meshkati, N. eds. *Human Mental Workload*, 1988, 139-183.
9. IDC. IDC Forecasts Worldwide Tablet Shipments to Surpass Portable PC Shipments in 2013, www.idc.com/getdoc.jsp?containerId=prUS24129713, 2013.
10. Kin, K., Hartmann, B., and Agrawala, M., Two-Handed Marking Menus for Multitouch Devices. *ToCHI* 18(3): article 16.
11. Kurtenbach, G. and Buxton, W. Issues of Combining Marking and Direct Manipulation Techniques. *Proc. UIST 1991*, 137-144.
12. Kurtenbach, G. and Buxton, W. User Learning and Performance with Marking Menus. *Proc. CHI 1994*, 258-264.
13. Kurtenbach, G., Fitzmaurice, G., Owen, R. and Baudel, T. The Hotbox: efficient access to a large number of menu-items. *Proc. CHI 1999*, 231-237.
14. Kurtenbach, G., Sellen, A. and Buxton, W. An Empirical Evaluation of Some Articulatory and Cognitive Aspects of Marking Menus. *Human-Computer Interaction* 8(1):1-23.
15. Kurtenbach, G.P. *The design and evaluation of marking menus*, PhD Thesis, University of Toronto, 1993.
16. Lepinski, G., Grossman, T. and Fitzmaurice, G. The design and evaluation of multitouch marking menus *Proc. CHI 2010*, 2233-2242.
17. Parhi, P., Karlson, A. and Bederson, B. Target size study for one-handed thumb use on small touchscreen devices. *Proc. Mobile HCI 2003*, 203-210.
18. Park, Y., Han, S., Park, J. and Cho, Y. Touch key design for target selection on a mobile phone *Proc. Mobile HCI 2008*, 423-426.
19. Roy, Q., Malacria, S., Guiard, Y., Lecolinet, E. and Eagan, J. Augmented letters: mnemonic gesture-based shortcuts *Proc. CHI 2013*, 2325-2328.
20. Scarr, J., Cockburn, A., Gutwin, C. and Bunt, A. Improving command selection with CommandMaps *Proc. CHI 2012*, 257-266.
21. Scarr, J., Cockburn, A., Gutwin, C. and Malacria, S. Testing the Robustness and Performance of Spatially Consistent Interfaces. *Proc. CHI 2013*, 3139-3148.
22. Serrano, M., Lecolinet, E. and Guiard, Y. Bezel-Tap gestures: quick activation of commands from sleep mode on tablets *Proc. CHI 2013*, 3027-3036.
23. Wagner, J., Huot, S., and Mackay, W. BiTouch and BiPad: Designing bimanual interactions for hand-held tablets. *Proc CHI 2012*, 2317-2326.
24. Wu, M., and Balakrishnan, R. Multi-finger and whole hand gestural interaction techniques for multi-user tabletop displays. *Proc. UIST 2003*, 193-202.
25. Zhao, S., Agrawala, M. and Hinckley, K. Zone and Polygon Menus: Using Relative Position to Increase the Breadth of Multi-Stroke Marking Menus. *Proc. CHI 2006*, 1077-1086.

Appendix C

Publication: Supporting Novice to Expert Transitions in User Interfaces

Supporting Novice to Expert Transitions in User Interfaces

ANDY COCKBURN, University of Canterbury
CARL GUTWIN, University of Saskatchewan
JOEY SCARR, University of Canterbury
SYLVAIN MALACRIA, University of Canterbury

Interface design guidelines encourage designers to provide high-performance mechanisms for expert users. However, research shows that many expert interface components are seldom used, and that there is a tendency for users to persistently fail to adopt faster methods for completing their work. This paper summarizes and organizes research relevant to supporting users in making successful transitions to expert levels of performance. First, we provide a brief introduction to the underlying human factors of skill acquisition relevant to interaction with computer systems. We then present our focus, which is a review of the state of the art in user interfaces that promote expertise development. The review of interface research is based around four domains of performance improvement: *intramodal improvement* that occurs as a factor of repetition and practice with a single method of interaction; *intermodal improvement* that occurs when users switch from one method to another that has a higher performance ceiling; *vocabulary extension,* in which the user broadens their knowledge of the range of functions available; and *task mapping,* which examines the ways in which users perform their tasks. The review emphasizes the relationship between interface techniques and the human factors that explain their relative success.

Categories and Subject Descriptors: **D.2.2 [Design Tools and Techniques]**–*User Interfaces;* **H.5.2 [User Interfaces]**–*Graphical User Interfaces (GUI)*

General Terms: Human Factors

Additional Key Words and Phrases: Expertise, novice to expert transition, shortcuts.

1. INTRODUCTION

Graphical User Interfaces (GUIs) mediate most communication between humans and computing devices. Their success is partly due to their natural support for novice users – the phrase 'see and point versus learn and remember' [Shneiderman 1987] describes how novices benefit from being able to visually find salient interface elements and manipulate them through a metaphor of direct manipulation. However, the very characteristics that make GUIs effective for novices also cause them to fail in their goal of supporting experts, and GUIs often trap users into a 'beginner mode' of operation. The richness and power of human perception, cognition, and motor action is constrained by standard GUI mechanisms such as visual search and direct manipulation, which are easy to understand but which force the experienced user into relatively slow and laborious action. Conversely, interfaces explicitly designed for experts (e.g., keyboard shortcuts or command-line interaction) allow high levels of performance, but only after extensive training. While the design of interfaces for novices *or* for experts has been well investigated, the design of interfaces that facilitate a transition from novice to expert performance is less well understood.

The problem of users failing to achieve expertise has been shown in many studies across a wide range of interactive tasks and contexts. In the area of Computer Aided Design, field studies demonstrated that long-term users often employ inefficient strategies for completing tasks [Bhavnani and John 2000], and log studies have shown surprisingly limited command vocabularies among experienced users [Matejka, Li, Grossman and Fitzmaurice 2009]. Similar findings have been shown for text editing [Rosson 1983], operating systems [Draper 1984; Doane, Pellegrino and Klatzky 1990], and spreadsheets [Nilsen, Jong, Olson, Biolsi, Rueter and Mutter 1993]. Furthermore, studies of interface mechanisms designed to facilitate expertise, such as keyboard shortcuts, have shown that these mechanisms are seldom used [Lane, Napier, Peres and Sandor 2005; Alexander 2009].

Carroll and Rossen [1987] used the phrase 'the paradox of the active user' to encapsulate the tendency for users' interface performance to reach an asymptote at a

level of mediocrity. They explained this effect through two biases that people bring to their work: a *production bias,* which encourages people to use known methods in order to quickly complete immediate tasks, rather than devoting time to finding better methods that may improve throughput in the long run; and an *assimilation bias*, in which people apply existing knowledge to interpret and solve new problems. Both biases lead users to continue with known methods of interaction, causing an associated tendency to miss opportunities for improvement. In scrutinizing this paradox, Fu and Gray [2004] additionally explained that preferred procedures for interaction are maintained over faster recommended alternatives when the preferred procedure is well-practiced and generally applicable, and when it provides fast, incremental feedback.

The tendency for users to asymptote at mediocre performance has serious and enduring implications for the productivity of millions of office workers. The tools used to conduct everyday office work, such as word processors and spreadsheet applications, have maintained substantially consistent point-and-click graphical user interface mechanisms across decades of interface revisions, and their expert methods of interaction, such as keyboard accelerators or 'hotkeys', have also remained relatively stable. Furthermore, recent interface releases have tended to reduce the visibility and salience of expert methods for interaction – for example, many hotkeys available in Microsoft Word have no visual depiction at all (e.g., Ctrl-Alt-M for "New Comment"), so a user who wants to become more efficient with these commands must resort to external sources, such as a web search, which may further reduce the likelihood of learning the expert method.

The typical design of graphical user interfaces, therefore, allows users to quickly learn suboptimal methods for task completion (e.g., by pointing and clicking), which users tend to maintain for months, years, and decades of subsequent interaction. While the productivity loss associated with each command selection is relatively small, these losses are multiplied across dozens of selections of each command per day (e.g., each invocation of the 'bold' command), for dozens of distinct commands, for several applications, for hundreds of days each year, and for dozens of years. The total productivity loss is analogous to an office worker who never learns to touch type – each keystroke is a minimal cost, but days of productivity per year are lost due to inefficient interaction methods.

Although many studies have demonstrated a tendency for users to fail to attain expertise, until recently there has been relatively little research into interface methods that can promote and assist the transition to expertise (with some notable exceptions, reviewed in Sections 4-7). Consequently, while designers know the problem exists, interface design guidelines provide little assistance. For example, Nielsen's [1993] influential usability heuristics give the abstract guidance 'Provide Shortcuts' (which prior research shows are unlikely to be used) and 'Help and Documentation' (often the user's source of last resort [Randall and Pedersen 1998]). Recently, however, there has been an increase in literature explicitly addressing the problem of how interfaces can be designed to promote and assist expert levels of performance. This paper provides a review and analysis of interface strategies that can be used to promote expertise, as well as providing a foundation in underlying human factors that can explain their success and point to promising new directions.

Naturally, learning and skill acquisition have been the focus of substantial research in the psychology literature. Texts such as Anderson's [2005] general review of cognitive psychology dedicate multiple chapters to the topic, and more specific texts such as Schmidt and Lee [2011] scrutinize how humans develop motor skills. The problem for interface developers and researchers, however, is that while there is abundant literature on skill development, there is a lack of clarity on how the findings can be applied in user interfaces to assist the development of user expertise.

This paper provides a review of literature addressing the problem of how user interfaces can assist users in transitioning from novice to expert levels of performance. The review is presented in two parts. First, we present a brief summary of key relevant findings on human skill acquisition, predominantly from the psychology literature. Second, we describe our focus, which is research from Human-Computer Interaction on interface techniques that are designed to support skill development, and we analyse their relative successes and failures. The review of interface research is divided into four sections, each addressing a different domain of performance improvement, progressing from low level issues of interface control through to high level issues of task strategy. The four domains are as follows: *intramodal improvement,* which occurs within a single interface method, such as when learning to operate a new pointing device; *intermodal improvement,* which occurs across interface methods, such as between mouse selection and hotkey use; *vocabulary extension,* which concerns the breadth of the user's knowledge of interface features; and *task mapping,* which concerns the way in which users approach their tasks, including how they learn their tasks and the strategies they employ. Finally, we identify a research agenda for future work on supporting expertise development with user interfaces. The overall aims of the paper are to summarize and distill existing knowledge on expertise development with user interfaces, to highlight successful strategies for assisting expertise development, and to motivate and direct further research on the topic.

2. HUMAN FACTORS OF LEARNING AND SKILL ACQUISITION

From an experimental-psychology perspective, learning and skill acquisition are often defined in terms of functions of memory [Schmidt and Lee 2011] – human memory is the repository for human experiences and understanding, so anything that has been learned is encoded in some form of memory. However, human memory is a vast research topic and a complete review is beyond the scope of this paper. Instead, we limit the review to summarizing the aspects of human memory that directly influence skill development with user interfaces – for example, spatial and proprioceptive memory strongly influence the user's ability to rapidly acquire interface targets, and consequently these functions are reviewed in Section 2.7.

Similarly, there is vast research literature on learning and education that is beyond the scope of this paper. As with memory functions, for brevity we limit coverage of this literature to summarizing key lessons that are applicable to skill acquisition with user interfaces. We refer readers seeking more general introductions to Anderson [2005] for general psychology, to Baddeley [1999] for a review of human memory, and to Thomas [2013] for an introduction to education.

This section focuses on underlying human factors of skill acquisition that are relevant to user interface design. It begins by describing three stages of skill development, and then reviews several factors influencing skill acquisition, including repetition, type and distribution of practice, the role of effort in efficient training, and how different forms of motivation and feedback affect training outcomes. Later sections explain how many of these human factors are employed in specific user interfaces that aim to improve users' transitions from novice to expert performance.

2.1 Stages of psychomotor skill development: cognitive, associative, autonomous

Various models of psychomotor skill acquisition have been proposed, with Fitts and Posner's [1967] three stage model being particularly influential. It describes how skills are developed through cognitive, associative, and autonomous phases, outlined in the following paragraphs. Learning to change gears on a manual car is a commonly-used example of this process, with novices at the cognitive phase forming initial models that might involve understanding the mechanics of a clutch and the

need to disengage the motor from the gearbox, removing power from the motor while doing so, etc. During the associative phase the learner focuses on repeating the necessary actions to refine their appropriate synchronization. And once autonomous, the coordinated actions are performed as a single burst of unconscious activity, leaving the driver's cognitive capacity free for higher level activities such as navigating or planning their day.

Cognitive phase

During the cognitive phase, initial conceptions of the activity are formed, predominantly learning *what* activities are to be done. Knowledge at this phase is substantially declarative and explicit, and might be communicated through verbal or written instructions, or through direct observation or visual images of the activity.

Performance of tasks at the cognitive phase of development is characterized by *controlled interaction*. Schneider and Shiffrin [1977] identified the following characteristics of controlled interaction: 1) it is slow; 2) it is attention-demanding, in that other similar tasks interfere with its execution; 3) it is serial in nature; 4) and it is strongly volitional, in that the activities can be avoided or immediately stopped. Task performance during the cognitive phase is also inconsistent, partially due to the learner's exploration of alternative strategies of what to do. Despite this variance, the magnitude of overall improvement in task performance is much larger during the cognitive phase than any other phase of learning.

The most effective training strategies during the cognitive phase focus on building up the person's explicit conceptualization or understanding of the task. They therefore typically involve explicit instruction, provision of clear models, or some other form of feedback that assists with task conception. Psychology research examining effective training methods for each phase are reviewed later in Section 2.8.

Associative phase

The associative phase (also referred to as the 'fixation' phase) is characterized by improvements in the motor actions used to execute the task. While the cognitive phase is dominated by attention to *what* is done, the associative phase is dominated by attention to *how* it is done. Performance improvements in this phase are smaller than the cognitive phase, but the variance in performance decreases. Where improvements are made, they generally involve subtle adjustments of execution. The verbal, declarative and explicit understanding of the execution of the task that characterized the cognitive phase is largely unused during the associative phase.

The associative phase is enduring, with many tasks requiring years or decades of practice before transitioning from the associative phase to automaticity.

Autonomous phase

The autonomous phase represents the ultimate level of psychomotor learning, which is attained after prolonged and extensive practice. Key characteristics of *automaticity* are the opposite of those characterizing the controlled interaction at the cognitive phase. Schneider and Shiffrin [1977] identified the following characteristics of autonomous actions: 1) they are fast; 2) they are not attention demanding, in that other verbal or cognitive operations do not interfere with their execution; 3) they are parallel in nature, with various operations possibly occurring together; 4) they are not volitional, in that processing is often unavoidable. Experienced touch-typists are likely to have encountered the non-volitional component of automaticity – when interrupted, a few pending keystrokes will be typed as a burst of activity prior to dealing with the interruption. Furthermore, there is evidence that people are unable to voluntarily terminate execution of these bursts [Salthouse 1985].

When an individual's task performance moves towards automaticity there is a reduction in the mental and physical effort expended on the task [Kahneman 1973].

This reduction can be observed using measures such as pupil dilation (pupils enlarge with mental effort, e.g., [Hyönä, Tommola and Alaja 1995]), galvanic skin response (skin conductivity increases with effort, e.g., [Engström, Johansson and Östlund 2005]), or fMRI (e.g., [Hasegawa, Carpenter and Just 2002]). Dual attention tasks are commonly used to examine automaticity because the parallel processing and absence of attention that accompanies automaticity allows improved performance on concurrent activities, with relatively small detriment to the skilled execution.

2.2 The Power Law of Practice

The three-phase model of skill acquisition provides a useful characterization of how people conceptualize and enact tasks, as well as suggesting how different training interventions might assist skill development at each phase (described further below). Although the phases are not discrete and different individuals may progress at very different rates, when aggregate task performance is viewed across time the performance curve follows a smooth and continuous power function, with substantial initial gains that gradually diminish. This aggregate 'power law of practice' has been reliably observed across a wide variety of tasks and studies, including Snoddy's [1926] early studies of drawing mirror images, Crossman's [1959] study of factory workers rolling cigars, and Card et al.'s [1983] study of text-editing tasks. The generality of this practice effect was noted by Fitts [1964] and named the 'power law of practice' by Newell and Rosenbloom [1981; 1981], although Heathcote [2000] argues that an exponential law is more accurate.

2.3 Type of Practice, and Individual Differences

Although the power law of practice has been shown to apply in many domains, there is strong evidence that repetition alone is insufficient for the attainment of elite levels of skill. Ericsson [2004] reviews multiple studies of elite performance across diverse domains (including musicians, athletes, and chess players), concluding that in addition to many thousands of hours of practice, the type of practice is critical to the attainment of true expertise. In particular, he demonstrates the necessity of *deliberate* practice, which focuses on tasks beyond the person's competence and comfort [Ericsson, Prietula and Cokely 2007]. Deliberate practice facilitates two kinds of learning – improving skills already attained, and extending the reach and range of skills; it is further assisted through expert mediation, such as skilled coaching.

Ericsson refers to the cognitive, associative, and autonomous phases of learning (Section 2.1) when explaining the difference between the attainment of 'everyday skills' and 'expert' (or elite) performance. As shown in Figure 1, he argues that *everyday skills* are developed in agreement with the description in Section 2.1, progressing through cognitive and associative phases, before ultimately allowing a satisfactory level of performance once autonomous. However, he contends that much higher levels of elite performance require deliberate practice to counteract automaticity during practice, with performers using various forms of mental representations to maintain the cognitive and associative phases during practice [Ericsson 2004], p. s4]. Further, he observes that once deliberate practice is abandoned, the level of elite performance may deteriorate.

2.4 Depth of Processing and Effort

The deliberate use of mental mechanisms to assist continued learning and improvement described by Ericsson are reflected by the research of Craik and Lockhart [1972], who proposed a "levels of processing" framework in memory research. They postulated that the strength of a memory is a positive function of the depth to which the stimulus is analyzed. Several subsequent studies validated the

framework – for example, Craik and Tulving [1975] showed that 'deeper' or more effortful mental manipulations during word memorization improved recall over 'shallow' encodings. In their experiment, word stimuli for memorization were coupled with shallow or deep questions, where 'shallow' encodings were based on word structure (e.g., "is the word in capital letters?") or on phonetic properties (e.g., "does the word rhyme with 'weight'?"), while 'deep' encodings were based on categories (e.g., "is the word a type of fish?), or on sentence fitting (e.g., "would the word fit in the sentence 'he met a _____ in the street'?").

Regardless of the exact framework used to encapsulate memorization procedures, there is now comprehensive empirical evidence that *elaborative processing* [Anderson 2005] (p.193), which increases the difficulty of processing stimuli for memorization, has a critical role in learning. This is an important and possibly counterintuitive finding for systems that are intended to improve ultimate user performance, because it suggests that interventions that cause temporary performance degradation during training may be beneficial in the long term – "manipulations that degrade the speed of acquisition can support the long term goals of training" (p.207, [Schmidt and Bjork 1992]). This issue is revisited in Section 2.9, which examines the *guidance hypothesis*.

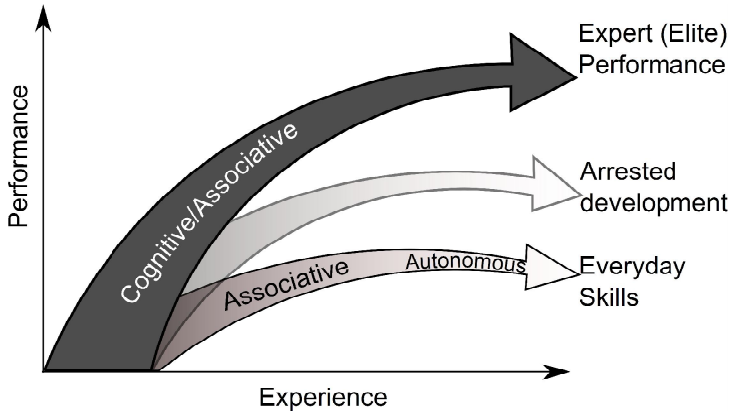


Figure 1. Ericsson's illustration (adapted from [Ericsson et al. 2007]). Everyday skills develop through cognitive and associative phases to reach a satisfactory level of performance once autonomous. Development of elite skills, in contrast, requires deliberate practice, to maintain the cognitive and associative phases required for improvement.

2.5 Convergence and Divergence of Individual Differences with Practice

In some conditions, the variance across different individuals' performance decreases following practice, but in others, the variance increases. Studies on the critical role of deliberate practice and on the importance of mental effort help explain why some people attain high levels of performance while others do not despite prolonged and intense repetition. As Ericsson puts it, "it may appear that excellence is simply the result of practicing daily for years or even decades. However, living in a cave does not make you a geologist" [Ericsson et al. 2007], p3.

Several researchers have further analyzed issues of individual differences in skill development. Ackerman provides a succinct summary of the conditions under which individuals' performance becomes more similar or more different as a result of practice [Ackerman 2007]. For simple tasks that involve speed and accuracy of motor movement, results suggest that individual differences in performance diminish as a result of practice. For example, the individual differences between people first using a mouse to control a cursor might be large, but performance is likely to converge with practice. However, if a motor skill task is more complex or if it enables unobvious styles/strategies of use, then performance differences may stay constant or diverge with practice. For example, two novice users of a keyboard may begin with a 'hunt-

and-peck' strategy of one-finger typing, but one user may dedicate deliberate practice to touch-typing, thus accelerating their performance, while the other does not.

As the touch-typing example suggests, as tasks become more complex or strategic, or as they demand more knowledge, there is a tendency for divergence to increase between the levels of highest and lowest performance.

2.6 Distribution of Practice

The effectiveness of training sessions can be influenced by altering their temporal distribution. Across many studies it has been reliably demonstrated that rest periods between training sessions improves learning in comparison to the absence of rests. Furthermore, longer rest periods appear to be more beneficial than short ones. These effects have been empirically demonstrated in the psychomotor domain for both short term activities (e.g., a point-following task with 30 second training batches and rests of between 0 s and 60 s, as used by Bourne and Archer [1956]) and long term activities (e.g., typing tasks trained once or twice daily for a total of 60 hours, as used by Baddeley and Longman [1978]). Similar effects have been demonstrated for cognitive learning. For example, Cepeda et al. [2006] present a review and synthesis of 184 papers studying verbal recall tasks, demonstrating that spaced (versus continuous) practice sessions improve recall performance. Their review also reveals an interaction between the optimal spacing period and the intended duration of memory retention, with longer spacing periods working best for longer retention.

2.7 Spatial and Proprioceptive Memory

Spatial memory is particularly important for efficient interaction with graphical user interfaces. This is because users can make rapid decisions about the location of target items when the interface maintains constant relative spatial locations, but when spatial predictability is compromised users must resort to comparatively slow visual search [Cockburn, Gutwin and Greenberg 2007; Cockburn and Gutwin 2009].

Spatial memory is a substantial research field within psychology (Anderson [2005], chapter 4), but the findings reported above also apply to spatial memory. For example, the *method of loci* can be considered to be a form of elaborative processing that aids recollection by associating memory stimuli with known spatial locations (such as the rooms in a familiar house). Related 'depth of processing' spatial memory effects have been reported in studies such as Naveh-Benjamin [1987] and Van Asselen, Fritschy and Postma [2005], as well as several studies specifically concerning interaction with computer systems (reviewed later in Section 4.2).

The input to spatial memory may be represented by visual, aural or proprioceptive stimuli. While graphical user interfaces are predominantly portrayed visually, consistent proprioceptive actions for control is likely to yield significant advantages for expert users. For example, the consistent spatial location of a vehicle's brake pedal (relative to the driver) facilitates rehearsal of precisely the same physical action (moving the foot leftwards and depressing the pedal) to slow the vehicle, with obvious advantages in an emergency. Touch-typing allows equivalent performance benefits for acquiring consistently defined proprioceptive targets in space. Conversely, however, the need for different movements to a target dependent on varied initial cursor location may impair the ultimate performance ceiling and ability to attain autonomous levels of performance with traditional mouse and cursor interaction (further explored later in Section 4.2).

2.8 Motivation and Feedback

Positive motivation to learn and improve is a key determinant in whether a learner chooses to participate in practice. Understanding how interventions influence

motivation is important in areas such as industrial psychology, sports science, and interface design for expertise.

Two key strategies for improving motivation are to make the task seem important and to use goal setting [Schmidt and Lee 2011]. These strategies are not distinct – for example, a person may view performance improvement as unimportant until stimulated by the presentation of a goal representing a higher performance that reflects the mean performance of others. Consequently, several studies have attempted to understand the types of goals that lead to the highest performance and learning outcomes. Interfaces using related methods to promote skill development are reviewed in Section 5.1.

In general, studies agree that specific, absolute goals of moderate difficulty yield better performance and learning outcomes than non-specific goals (such as "do your best") or no goals [Tubbs 1986; Kyllo and Landers 1995; Locke and Latham 2006].

Motivational information can be considered to be a form of feedback to the learner. The design of appropriate feedback mechanisms is a substantial component of interface development, so understanding the impact of different feedback methods on skill acquisition would greatly assist interface designers.

Naturally, there is extensive literature on the effectiveness of different forms of feedback in assisting learning in the cognitive domain, which includes abundant literature in the field of education. High-level guidelines emerging from these works are necessarily broad and abstract, such as 'create a respectful, friendly, open-minded and unthreatening climate', 'base feedback on observed facts', and 'suggest ideas for improvement' (e.g. [Hewson and Little 2001]).

In the psychomotor domain, where feedback is applied to learning physical movement, there is a closer relationship between psychology findings and their potential deployment in user interfaces (where users execute motor actions to activate particular interface functions). Schmidt and Lee [2011], chapters 11 and 12, provide a comprehensive review, covering both the *intrinsic* feedback effects (such as visual or proprioceptive stimuli) that are inherently coupled with limb movement, as well as the effectiveness of different forms of *extrinsic* feedback that can be explicitly applied to assist skill acquisition. Many parameters of augmented feedback can be manipulated, including the following:

- *temporal properties*, such as providing feedback that is either concurrent with the action, presented on termination of the action, or delayed;
- *aggregation,* such as providing independent feedback about each *discrete* action versus *accumulating* information about a sequence of actions;
- *modality* and *form*, such as textual or spoken instruction, video of the action, or statistical summaries of performance;
- *knowledge of performance* (KP), which concerns information about the way in which the action was executed, such as the deviation from the ideal movement. For example, a golfing tutor might show a video of a student's swing, possibly commenting on foot position or backswing speed. Note that the feedback concerns the movement, not its outcome.
- *knowledge of results* (KR), which concerns information about the outcome of the action that is presented after its termination, such as its success or failure, or the time taken to complete it. For example, a typing application might provide feedback in the form of a beep to indicate a target phrase or character was typed incorrectly, or it might show a plot of word-per-minute typing rate; similarly, a golfing tutor might note the distance that the ball travelled. Note that the feedback concerns the outcome, not the movement itself.

The distinction between *knowledge of performance* (KP) and *knowledge of results* (KR) is an important one that has been extensively studied in the psychomotor literature. Gentile [1972] (who introduced the term 'knowledge of performance')

postulated that KP is most effective for *closed* motor tasks, in which the task is uniformly performed without response to changing external factors, and where the movement itself is the goal of the skill – for example, a golf swing would be considered to be a closed motor task*. It is debatable whether certain forms of KP alone (such as non-augmented videos of performance) can yield substantial performance benefits – for example, Rothstein and Arnold [1976] showed that performance videos alone did not improve the participants' performance, possibly due to a lack of specificity about the actions required to improve. However, it is clear that performance and learning outcomes are enhanced when KP is combined with KR (e.g., Wallace and Hagler's study of basketball players [1979]). As with cognitive feedback, given the diversity of literature and their results, we defer details until reviewing interface methods in Section 4.2.

2.9 Guidance

Physical guidance is a form of feedback where the learner's limbs are externally manipulated in an attempt to assist learning physical movements. Results vary on the learning efficacy of external guidance. For example, Armstrong [1970] showed that for learning complex elbow movements, mechanical limb control resulted in precise movements during training, but inaccurate reproduction of the movement once the controls were released. In contrast, when training was achieved without mechanical control but with visual feedback of a motion plot (either at the end of the trial or concurrently during it), participants produced the motions most accurately (once feedback was removed) when trained in the post-movement feedback condition. In other words, delayed and unconstrained feedback achieved the best learning outcome. Not all experimental results agree with this finding, and it seems likely that some form of coarse physical guidance may assist learners in their initial conceptualization of the required movement, particularly if the movements are large and relatively slow (e.g., teaching a child a kayak stroke).

These results also emphasise important questions about the methodology used for evaluating skill acquisition and learning. In particular, the *guidance hypothesis* [Schmidt 1991] suggests that augmented feedback which improves early performance through guidance may impair retention of the performed skills once the guidance is removed. Consequently, most experiments in the psychomotor domain use separate experimental periods for *training*, *retention* (the trained tasks are evaluated later, often 24 and 48 hours after initial training), and *transfer* (where the skills are applied to a related but non-identical task).

While it may seem unclear that physical guidance applies to user interfaces, force-feedback and tactile user interfaces are increasingly being examined as potential training aids for tasks including dentistry procedures [Rhienmora, Gajananan, Haddawy, Suebnukarn, Dailey, Supataratarn and Shrestha 2010] and teaching the blind to write signatures [Plimmer, Crossan, Brewster and Blagojevic 2008]. Furthermore, many user interfaces exploit some form of dynamic visual guide, or 'feedforward', to assist users in performing their task. We return to specific user interface examples in Section 4.

2.10 Other human phenomena influencing skill acquisition

The literature summarized above seeks to understand how humans respond to the manipulation of various factors that influence learning and skill development. Although many of the effects observed are empirically consistent and reliable, there

* Note that categorising tasks as 'open' or 'closed' is a different concept to open- and closed-loop motor control. The terminology is unfortunate because an open task requires closed-loop control.

are also various human phenomena that complicate the deployment and analysis of interventions intended to aid transitions to expertise.

Satisficing

Rather than seeking to continually optimize performance, people have a tendency to 'make do' with solution strategies that were first learned, even though they may be known to be suboptimal. Furthermore, once a suboptimal strategy has been learned and reproduced several times, it is likely to become habit, further reducing the likelihood of changing to faster alternative methods. Simon [1959] used the term 'satisficing' to describe how decision-makers often lack the information and cognitive resources needed to make a rational optimal decision, and also that once the costs of calculating a rational near-optimal decision are accounted for there will be little difference between attempts to optimize and making a faster, approximate 'satisficed' determination. In user interface research, satisficing has been used to explain phenomena such as users failing to make more extensive use of keyboard shortcuts despite their using substantially the same interface for years or decades (Section 5.1).

Gray and colleagues have extensively investigated the seeming paradox of prolonged suboptimal performance. Fu and Gray [2004] describe the tendency for users to persist with interaction methods that are well-practiced, are generally applicable, and which provide incremental feedback. Gray et al. [2006] also present evidence supporting their 'soft constraints hypothesis', which explains how short-term interactions (of duration between ⅓ and 3 seconds) can be construed as locally optimal within those timeframes, although globally suboptimal. In explaining this seeming contradiction, they use the analogy of a person following instructions to build a toy – the globally optimal solution might involve memorizing the full set of instructions before assembling any components, but a locally optimal one might intersperse reading instructions with assembly.

Arousal, personality and culture

The efficacy of certain training interventions has been demonstrated to interact with individual and environmental factors, which greatly complicates the process of determining which training methods are likely to be most effective for different people in different settings. For example, while it is known that performance generally follows an inverted 'U' shape with level of arousal (i.e., performance deteriorates when a person is under- or over-excited), there is evidence that personality differences such as introversion and extroversion result in different baselines of arousal, and consequently the same stimulus effect may increase performance for an extrovert while decreasing performance for an introvert [Revelle, Humphreys, Simon and Gilliland 1980; Bullock and Gilliland 1993]. Revelle et al.'s study also showed interactions between several other factors, including time of day, other personality factors, and consumption of stimulants.

Cultural differences may also play a role in determining the effectiveness of learning interventions. For example, Dahlin and Watkins [2000] used a qualitative method to describe the role of rote repetition in memorization and understanding among German and Chinese students. Their findings suggested that German students perceived rote lessons as being relatively ineffective 'surface' instruction, whereas Chinese students referred to benefits arising from the attentive effort required to discover new meanings in the materials studied.

Interaction with skill complexity

Several studies have shown an important interaction between the effectiveness of training interventions and the participants' skill level and the complexity of the skill conducted. Perkins-Ceccato, Passmore and Lee [2003] showed that low-skill golfers benefited more from instruction that focused attention on internal motor aspects of actions when compared to instructions that focused on external effects on the ball;

conversely, highly-skilled golfers benefited more from externally focused instructions. Similarly, Wulf and Shea [2002] found that principles derived from the study of simple tasks do not generalize to more complex skills. They explained this effect with reference to the learner's total cognitive load– when task demands are low, learners benefit from practice conditions that increase total load, whereas when task demands are high, practice conditions with more manageable workload are beneficial.

2.11 Summary of Human Factors of Skill Acquisition

Highly skilled task performance is characterized by automaticity, where the physical actions required for the task are executed quickly, efficiently, and with minimal conscious deliberation. Importantly, automaticity allows the performer to think about higher level tasks during the execution of the skill. While high degrees of repetition are required for skilled execution to approach automaticity, different training interventions influence the way performers gain skill and understanding.

This section reviewed some of the key human factors influencing psychomotor skill acquisition, with a focus on those most relevant to interaction with computer systems. As stated earlier, the aim was to provide a brief introduction to assist in understanding the objectives and methods of many of the interface techniques (described in upcoming sections) that have been used to encourage and support users in becoming more expert.

3. FOUR DOMAINS OF INTERFACE PERFORMANCE IMPROVEMENT

The following sections review prior research examining user interface methods that assist users in transitioning from novice to expert levels of performance. Typically, 'high performance' will be exhibited by rapid task completion, but in some interactive contexts the focus of improving performance may be on error reduction, or on increasing the quality of the product. Our focus is on methods that can be deployed *within* the user interface, rather than the broader range of methods that are external to the interface, such as offline training courses, the role of socialization in interface learning, or individual preference for learning style.

The review is structured by considering four domains that characterize different opportunities for improving performance with user interfaces. These domains are derived from a conceptual deconstruction of interaction with an interface as consisting of the user's *performance characteristics* with a set of *alternative methods* for activating a set of *functions*.

Functions are the set of interface commands and capabilities that allow data, state, or view manipulations to be achieved. For example, a simple painting application may support functions for drawing lines, painting shapes, zooming, scrolling, and so on. Analyses of functionally rich interfaces have shown that even experienced users typically use only a small subset of available functions [Draper 1984; Matejka et al. 2009].

Each interface function can typically be accessed through more than one interface *method*. For example, a line-drawing function might be selected by clicking a palette icon, through pull-down and context menus, or via a keyboard shortcut; and a scrolling function might be controlled by dragging a scroll thumb, by rotating a scrollwheel, or by pressing keyboard arrow keys.

Each *method* has associated *performance characteristics*, including a performance *floor* and *ceiling*, and each user will have attained performance ability somewhere between the floor and ceiling.

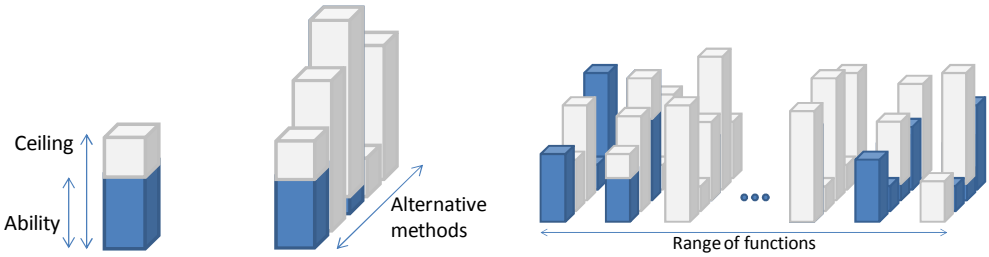
The four domains of interface performance improvement described in the following subsections are as follows:

1. *intramodal improvement* concerns the rapidity and magnitude of performance improvement with one particular interactive method (e.g.,

pointing with the mouse) for one particular function (e.g., selecting the bold function in a word processor);

2. *intermodal improvement* concerns ways to assist users in switching to faster methods for accessing a particular function (e.g., switching from cursor-based interaction to keyboard shortcuts);
3. *vocabulary extension* considers ways to help users broaden their knowledge and their use of the range of functions available in an interface;
4. *task mapping* addresses higher-level issues of the strategies that users adopt when seeking to complete their tasks with a user interface.

Figure 2 visually characterizes the key objectives of intramodal improvement, intermodal improvement, and vocabulary extension, as well as their relationship with interface methods and functions. The height of each column in Figure 2a depicts the performance ceiling enabled by a particular interaction method for a particular function, and the blue shaded region depicts the performance level attained by the user (to reiterate, this characterization is not dependent on any particular measure of performance). Figure 2b depicts a range of alternative methods for one function, emphasizing that different methods have different performance characteristics. Figure 2c depicts the range of functions available in a user interface, showing that users may be unaware of certain functions.



(a) Performance characteristics of a method. *Intramodal improvement* concerns the user's performance ability within one method.

(b) Range of methods for a function. *Intermodal improvement* concerns the user's transition to faster methods for activating a function.

(c) Range of functions available. *Vocabulary extension* concerns the user's knowledge and use of the functions available.

Figure 2. Conceptual representation of a user interface: **a**. methods that have distinct performance characteristics; **b**. alternative methods for any function; and **c**. a range of functions.

4. DOMAIN 1: INTRAMODAL PERFORMANCE IMPROVEMENT

Based on Newell and Rosenbloom's [1981] power law of practice, Figure 3 depicts how user performance improves with experience of a *single* interaction modality. In the description below, we subdivide this curve into three segments for *initial performance*, *extended learnability*, and *ultimate performance*. These three stages are suggestive of Fitts and Posner's [1967; Anderson 1995] *cognitive*, *associative*, and *autonomous* stages of skill acquisition, described in Section 2.1.

The following subsections describe factors influencing each of these stages, including the observation that some factors assisting performance at one stage can impair performance at another.

4.1 Initial performance

Interface design for the initial stages of learning is strongly promoted in most usability guidelines [Norman 1983; Norman 1983; Shneiderman 1992; Dix, Finlay, Abowd and Beale 1993; Nielsen 1993]. At this stage, users are unfamiliar with the interface, and must rely on their prior experiences, visual search, and recognition to find the commands they need. To optimize initial performance, designers aim to make interfaces easy to comprehend, with as high a performance floor as possible.

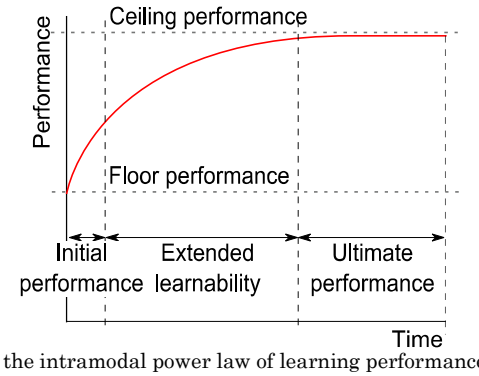


Figure 3. Characterisation of the intramodal power law of learning performance curve across time of use for one particular interface method for accessing one particular function.

Without prior experience to draw on, novice users are heavily reliant on visual search, and the time for visual search is a linear function of the number of candidate items that must be considered [Hornof and Kieras 1997; Wolfe 1998; Cockburn et al. 2007; Cockburn and Gutwin 2009]. Novice performance, therefore, can be improved by reducing the number of controls displayed to the user. Carroll and Carrithers [1984] exploited this effect with their 'Training Wheels' interfaces, which intentionally reduced the number of commands displayed. Related ideas were pursued with 'multi-layer interfaces' [Shneiderman 2003], which increase the level of functionality and reduce instructional interventions as the user becomes more experienced. However, by presenting different interfaces to novice and expert users, there are risks of confusing users when making a transition to the more advanced user interface (introducing a 'performance dip', as discussed in Section 5).

'Ephemeral Adaptation' [Findlater, Moffatt, McGrenere and Dawson 2009], shown in Figure 4, demonstrates a different approach to helping users quickly identify the most likely commands, but without requiring a substantially different interface presentation for novices and experts. With this technique, a small subset of command elements are immediately shown when the containing item is displayed, while the remaining items are gradually faded into view over a few hundred milliseconds. For example, when a menu is posted, the 'core' or most important functions can be immediately shown, while the remaining items are gradually exposed over time.

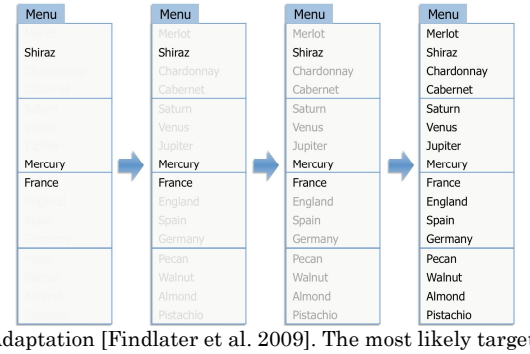


Figure 4. Ephemeral Adaptation [Findlater et al. 2009]. The most likely target commands are shown immediately, while others are faded in over a few hundred milliseconds. This technique can be used to draw a novice's attention to the most likely commands, while maintaining the original menu layout.

The notion that controls should be visible to be learned is also well expressed in most usability guidelines, but the corollary of making novice functionality visible is that expert functions are often suppressed, reducing the likelihood that they will be discovered. A related concept is that appropriate interface controls should be 'ready to hand' [Karat, Karat and Ukelson 2000]: that is, controls and feedback should be available for use but not obstruct task completion. Dyck et al. [2003] observe that

many computer games achieve the dual objectives of availability without obstruction through 'calm messaging' that uses presentation methods such as transient text, animation, and audio to provide information in a non-abrupt and non-intrusive manner.

4.2 Extended learning

Several factors influence the rate at which performance increases after initial familiarization, including the effort required during training and the use of guidance to assist performance. These factors also apply during *inter*modal improvement.

Effortful learning

In seeking to exploit findings from psychology suggesting that "deeper" mental encodings result in stronger memories [Craik and Lockhart 1972; Schmidt and Bjork 1992] (Section 2.4), several interface researchers have examined interfaces that improve skill development. For example, Cockburn et al. [2007] examined the effects of a 'frost brushing' interface for helping users to learn two things: first, the spatial location of letters in a modified keyboard layout; and second, the spatial shape of touchscreen gestures that correspond to entire words in ShapeWriter [Zhai and Kristensson 2003]. As Figure 5 shows, when using the frost brushing interface, the visual guidance normally provided by the display of keys on the keyboard was occluded by 'frost' over the display. To temporarily see the key locations, users had to brush away the frost by waving the stylus cursor over each key, with the frost quickly reforming. The design intention was to elevate deeper mental encodings by enforcing the need for memorization, instead of the normal continual availability of simple visual search.

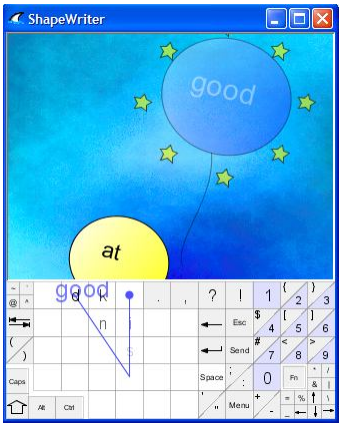


Figure 5. The 'frost brushing' training game for ShapeWriter [Zhai and Kristensson 2003], extracted from [Cockburn et al. 2007]. Letters on the keyboard are occluded by 'frost' that quickly reforms after being 'scrubbed away' with the stylus. A balloon bursts when its word-gesture is correctly entered.

Experimental results showed that the frost brushing condition improved both letter and shape memorization, compared to the traditional keyboard display. However, the participants' comments revealed design subtleties in choosing an appropriate level of effort. For letter learning tasks, the frost brushing condition was rated as more engaging and less frustrating than the normal interface; but for word shape learning tasks the frost brushing interface was rated as less engaging and more frustrating than normal. These results suggest that there is a fine balance between engaging and frustrating users, and that training interfaces should be neither too mundane nor too difficult. This finding reflects those of Wulf and Shea [2002] who observed that principles derived from the study of simple tasks may not generalize to more complex skill learning (Section 2.10).

In examining interface methods that support the development of touch-typing skills, Yechiam et al. [2003] proposed using a secondary task during text entry that imposed a 'moderate and immediate punishment' for looking at the keys rather than the text output. While entering text, the secondary task required a rapid response to the periodic display of a blue square in the text output area. If the user failed to quickly respond to the blue square's appearance, then the entire display would be temporarily dimmed. Users were much more likely to fail to notice the blue square if looking at the keys rather than the display. Experimental results confirmed that the inclusion of the secondary task improved participants' touch-typing performance by encouraging users to rely on key memorization (and look at the text output area), rather than focus on the keyboard.

A recent study by Kim and Ritter [Kim and Ritter] investigated learning, forgetting, and re-learning a spreadsheet task when using mouse-and-menu interaction methods in comparison to keyboard-command interaction. Their findings suggest a tendency for mouse-and-menu users to forget methods sooner than the keyboard-command group. This finding is consistent with the depth of processing hypothesis, as suggested by Kim and Ritter's characterization of mouse-and-menu interaction as relying predominantly on 'knowledge-in-the-world' in contrast to keyboard-command interaction relying on 'knowledge-in-the-head'.

Revelation, guidance, and rehearsal in marking menus

Kurtenbach et al.'s research on marking menus makes several substantial contributions on supporting natural transitions from novice to expert levels of performance with user interfaces [Kurtenbach 1993; Kurtenbach and Buxton 1993; Kurtenbach, Sellen and Buxton 1993; Kurtenbach and Buxton 1994; Kurtenbach, Moran and Buxton 1994]. The work lies on the cusp between intra- and inter-modal improvements because although marking menus support both novice and expert modes for interaction, the user's motor actions in both modes are intentionally very similar.

Figure 6, extracted from Kurtenbach et al. [1994], shows both the novice (left) and expert (right) modes for hierarchical menu item selection when using a marking menu. Marking menus are a form of radial menu, in which directional movements from a starting location select different items. Shortly after the user's stylus makes contact with the surface (Figure 6, left) a radial menu is displayed; the user moves northward to select the 'Groceries' submenu, causing a sub-menu to be displayed; and moving rightwards then selects the 'Fruit & Veg' item. Alternatively, an expert user (shown in Figure 6, right) can make a rapid gesture of similar shape to quickly select the same item, but without needing to pause for guiding feedback.

Marking menus were designed to adhere to a set of design guidelines, including *revelation*, *guidance*, and *rehearsal* [Kurtenbach et al. 1994], as follows:

Revelation is used to provide interactive feedback to the user about the commands that are available and how the user can invoke them. Revelation is particularly important for gestural interaction, where the user's actions need not have any associated representation on the display. Unlike buttons, scrollbars, and other graphical widgets that afford a particular method for activation (clicking, dragging, etc.), gestural interactions need not be conducted with reference to specific graphical entities. The *revelation* mechanism used by Kurtenbach et al.'s marking menus was 'press and wait for more information' – in other words, a dwell timeout was used before graphical feedback of the radial menu items was presented.

Guidance as a design principle states that the methods used for revelation should assist the user in specifying the complete command. In particular, guidance should not interfere with the user's specification of intention. In the context of a hierarchical marking menu, the appearance of a second-tier radial menu part way through a

gestural command (such as 'Meat' to 'Staples' in Figure 6) serves as *guidance*, while a pop-up dialogue box that removes attention from the task would not.

Rehearsal states that the way that guidance is provided should require a physical rehearsal of the expert's actions. In other words, the novice's motor actions should rehearse the expert's motor actions in order to assist the development of automaticity and muscle memory.

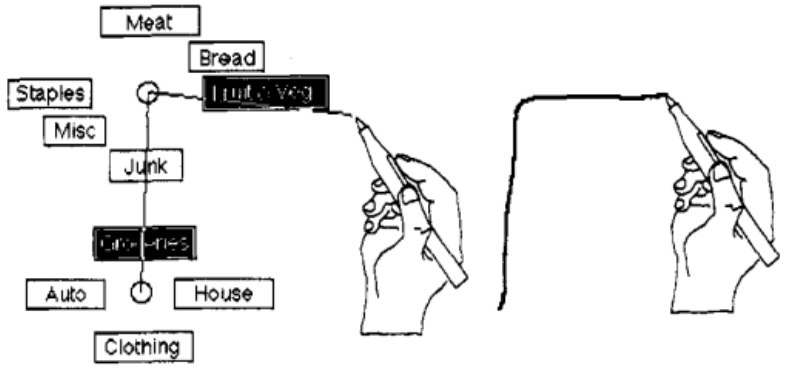


Figure 6. A hierarchical marking menu selection being made in response to visual guidance (left), and without visual feedback using a previously rehearsed action (right) [Kurtenbach et al. 1994].

Guidance versus effort for attaining performance versus learning

Kurtenbach et al.'s marking menu research was conducted in the early 1990s, with a focus on stylus-based input. The recent surge in popularity of finger-based gestural interaction on mobile devices has renewed interest in advanced techniques for easily learned and high performance gestural interaction techniques.

'OctoPocus' [Bau and Mackay 2008], shown in Figure 7, uses dynamically adapted guidance to assist the user in performing and learning the set of gestures available – the technique assists with both the performance of specific gestures (intramodal improvement) as well as learning other gestures that have similar initial gestural shapes (vocabulary extension). When the finger (or object) makes contact with the surface, the available gestures are shown, with the initial paths emphasized and more distant path components faded. Once the user begins tracing the gesture, the 'feedforward' portrayal of the required remaining gestural action adapts to the completed gesture components – for example, the path for 'Paste' is removed in the right-hand image of Figure 7, and the path for 'Cut' is deemphasized by making its path thinner. OctoPocus was evaluated in comparison to traditional Help menus and marking menus, showing positive results. Related systems addressing similar concepts of real-time guidance include the gesture prediction methods of Appert and Bau [2010] and Kristensson and Denby [2011], also demonstrated by interactive systems SimpleFlow [Bennett, McCarthy, O'Modhrain and Smyth 2011] and ShadowGuides [Freeman, Benko, Morris and Wigdor 2009].

In recent work, Anderson and Bischof [2013] conducted experiments to determine whether Schmidt's *guidance hypothesis* [1991] applies with guidance-based interactive methods (Section 2.9). They compared user performance using four different interfaces for providing guidance: 'crib notes', which provided static representations of the gestures, continually displayed in the corner of the display; 'static-tracing' representations, which showed non-responsive versions of all gestures at the starting location of the gesture; 'dynamic-tracing', equivalent to OctoPocus; and 'adaptive-tracing', which was identical to the 'static-tracing' condition except that the guide disappeared at a progressively earlier point through the training set (disappearing at the end of the gesture in the first training event, and at the start of the gesture in the last training event). Importantly, their experimental method

analysed both performance during training, as well as learned retention of the gestures, measured 24 hours after the training session.

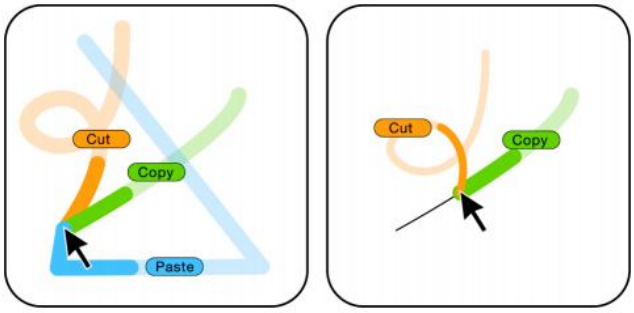


Figure 7. Dynamic gestural guidance with OctoPocus [Bau and Mackay 2008]. Initially (left), three alternative gesture commands are possible, but as the use gestures upwards and rightwards, guidance for 'Paste' is removed and 'Cut' gradually fades, leaving only the remaining guidance for 'Copy'.

Their results confirmed the guidance hypothesis – the conditions that caused the highest levels of performance during training (static- and dynamic-tracing conditions) performed the worst in the retention tests, and vice versa.

These results reflect those generated by Cockburn et al.'s [2007] frost-brushing experiments, in which the frost-brushing condition slowed user performance during training, but resulted in better results during subsequent testing conditions. Similar results have been generated for *intermodal improvement*, such as Grossman et al.'s [2007] study of hotkey learning, described in Section 5.1.

These results have important implications for the design of training systems, as designers must understand whether their goal is to assist users in rapidly attaining high performance, or whether learnt outcomes and the ability to transfer the learning to related interactions are primary objectives.

4.3 Ultimate performance

The final characteristic of the intramodal curve is the asymptote, or performance ceiling (Figure 3). There is extensive literature on supporting and understanding expert interface performance (although it is largely independent of the processes of enabling its attainment). In particular, Card, Moran and Newell [1983] provide models and empirical evidence of 'expert performance of routine tasks', including analysis of one user who repeated the same editing task thousands of times to study the progression to automaticity. Four interface characteristics that contribute to high performance ceilings follow.

Flat command structures

GUIs typically contain more controls than can be easily displayed at once, necessitating interface partitions such as windows, tabs, and menu hierarchies. Navigating through these partitions takes time, and consequently there are potential performance benefits in flattening the command structure to make more items accessible at once [Scarr, Cockburn, Gutwin and Bunt 2012]. Commands issued by command line interfaces (CLIs) and hotkeys are exemplars as they have global interface scope (e.g. <Ctrl>-C executes 'copy' regardless of the interface state). Several research and commercial systems have used CLIs to improve interface performance: e.g., Quicksilver[†], Spotlight[‡], Enso[§], and GEKA [Hendy, Booth and

[†] http://www.blacktree.com
[‡] http://support.apple.com/kb/HT2531
[§] http://humanized.com

McGrenere 2010]. Although empirical results for CLI benefits over GUIs have been mixed (e.g. Whiteside, Jones, Levy and Wixon [1985]), it is widely accepted that CLIs enable higher efficiency, and power users are strong advocates (e.g. [Barrett, Kandogan, Maglio, Haber, Takayama and Prabaker 2004]).

Terse and expressive input
Powerful interface commands communicate a lot of meaning in rapidly expressed actions [Scarr, Cockburn, Gutwin and Quinn 2011]. For example, a single alphabetic character can discriminate 26 commands, or 52 with case sensitivity; increasing to 2704 with two case-sensitive characters. However, there is often a tension between supporting terse, expressive power and meaningful mappings: for example, Alt-shortcuts in Office 2007 allow access to most interface controls, but they are often abstract and hard to remember (e.g, '<Alt> n, nu, t' inserts a page number in Microsoft Word).

Revisitation/history support
Users' interactive behaviour is often repetitive (e.g., command use [Greenberg and Witten 1993] and web navigation [Tauscher and Greenberg 1997]), and interfaces can aid efficiency by explicitly supporting command repetition. For example, web browser URL address bars and the Google search box memorize previous activities and offer type-ahead shortcuts for them: e.g., the keystrokes 'cn<Return>' become a shortcut for a user who frequently visits CNN's website. Like marking menus, such interactions lie on the cusp with intermodal transitions, but we categorize them within intramodal improvement because the initial mechanisms for interaction are identical for both novice and expert modalities.

Spatial predictability
Studies have demonstrated that spatial stability allows users to make rapid decisions about items' locations rather than relying on comparatively slow visual search (e.g. [Cockburn et al. 2007]). Despite the desirability of spatial stability, this principle is often compromised due to display space constraints or to reconfiguration of the display – interface controls are often elided and repositioned as window geometry is manipulated, and this is necessary because widgets typically do not scale. However, this is a technical, rather than a human, limitation: Scarr et al. [2013] examined the robustness of users' spatial memory to various forms of spatial transformation (including scaling), finding that users were able to quickly locate items in spatially stable views that had undergone a uniform spatial transformation.

In their 'InfoCockpit' project, Tan et al. [2001] examined whether items displayed on multiple monitors wrapping around the user would enhance interaction based on the spatial predictability of items. Spatial audio was included to assist memorization of place, and their results showed a 56% increase in object memory compared to standard desktop systems.

5. DOMAIN 2: INTERMODAL PERFORMANCE IMPROVEMENT

Most interfaces support more than one interaction method for accessing the same function (Figure 2b). Once a user approaches their intramodal performance asymptote, any further performance improvement requires shifting from one method to another that offers a higher performance ceiling. For example, a user's performance in cursor selection of a word processor's bold button will improve as they learn the location of the icon, but substantial further performance improvement is possible if the user switches to a different method for issuing the bold command, such as using the associated keyboard shortcut. Intermodal performance improvement concerns these transitions between interface methods.

However, Scarr et al.'s [2011] analysis of *intermodal* performance improvement postulated the existence of an important 'performance dip' that occurs when the user

switches to another method, as depicted in Figure 8, which extends beyond the intramodal considerations of Figure 3. The existence of this performance dip is likely to deter users from switching to new methods for interaction, even though doing so may offer long term productivity gains – the prospect of encountering a short term productivity loss may cause users to postpone switching to the new method, possibly forever. Factors acting as deterrents or barriers to intermodal transitions, and methods to overcome them, are reviewed in the following subsections, which address two critical points on the *inter*modal performance curve shown in Figure 8: first, factors influencing the initial switch to a new interface modality; and second, the performance dip that a user is likely to experience when switching from a familiar interface to an unfamiliar one.

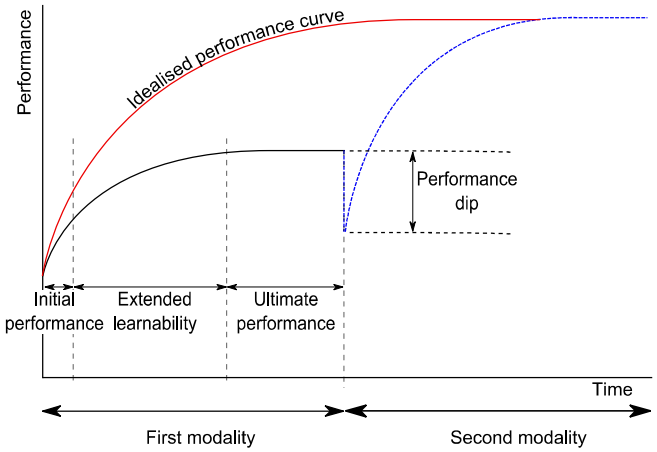


Figure 8. Performance curves characterising improvement within one interface method (left, 'intramodal improvement') and the performance dip that occurs when switching to a new, higher performance ceiling, method (right, 'intermodal improvement'). Adapted from Scarr et al. [Scarr et al. 2011].

5.1 Making an initial switch

Awareness of the new modality

The first stage of supporting a transition to an alternative method for interaction is to make the user aware of the modality. However, this awareness can be achieved through a wide variety of means, with varying impact and effectiveness. For example, traditional menus typically display the keyboard shortcut associated with each menu item, although users may not attend to them; and toolbar items often display rollover tooltips that include the hotkey, although a dwell timeout prior to displaying the tooltip reduces the likelihood that the user will encounter the hotkey during normal interaction.

Interfaces can be forceful in their awareness mechanisms, requiring users to experience the new modality by demanding that actions are completed using it. Grossman [2007] experimented with a variety of schemes for assisting hotkey learning. These included visual and audio schemes to expose users to the hotkeys, a delay-based technique to deter use of the GUI (i.e., making the system unresponsive for 2 seconds after each selection), and a technique that forced hotkey use after each menu selection. Their results showed that forced use and audio feedback worked well, with 72.8% and 66.6% of experimental selections being made with hotkeys. Subjective data showed no significant adverse response to the audio and forced use. In similar work, the HotKeyCoach [Krisler and Alterman 2008] revealed a dialog box whenever an item was selected with the mouse, requiring either an extra click to proceed, or the hotkey to be issued. Although results show these methods to be successful in increasing hotkey use, they may be unacceptable to users.

An alternative approach to awareness was demonstrated by the 'calm notification' method employed by Scarr et al. [2011]. Their system, called 'Blur', aimed to promote a transition from novice point-and-click interaction to higher performance command-line interaction by supporting command-line control of interactive components within unaltered commercial GUI interfaces. To help users develop awareness of the alternative command language, their calm notification method used a transient and transparent window which would briefly appear in the corner of the screen showing the name of the command. Figure 9a shows the calm notification window overlaid on the Microsoft Word ribbon following selection of the Align Left toolbar icon ( ), and Figure 9b shows Blur's command recommendations after the user types '<Esc> al' – hitting the enter key selects the top-most selection.

(a) Blur's calm notification of the command line alternative ('Align Left') for the GUI selection of the toolbar item. The window disappears after 1 second and can be clicked through so as to not prevent pointer-based interaction with underlying widgets.

(b) Command recommendation and command line completion in Blur. The user has typed 'al', and the first predicted command is 'align left', which is selected by pressing Return.

Figure 9. Blur's transient and transparent user interface.

Perception of the new modality

Once the user is aware of an alternative modality, the probability that they will switch to using it is influenced by their perception of its future efficiency, so all of the intramodal factors described above play a role. Importantly, though, several studies have demonstrated that a user's predicted future experience differs from their actual experience (e.g. [Czerwinski, Horvitz and Cuttrell 2001]), and that users can mistrust their abilities, leading to under-estimates of potential benefit with the new modality. For example, Cockburn and McKenzie [2002] showed that users predicted that they would perform poorly in a spatial task, but rated their actual performance much higher. Similarly, studies have consistently shown that keyboard shortcuts offer a higher performance ceiling than mouse selection [Odell, Davis, Smith and Wright 2004; Lane et al. 2005; Malacria, Bailly, Harrison, Cockburn and Gutwin 2013], yet Tak et al. [2013] found that some participants did not use known hotkeys because they believed the toolbar buttons to be faster.

Even when users are cognizant of the benefits of switching to a new modality, they may not do so due to the 'paradox of the active user' in which users "are likely to stick with the procedures they already know, regardless of their efficacy" [Carroll and Rossen 1987]. In seeking to resolve this paradox, Fu and Gray [2004] examined three substantially different tasks, forming cognitive models of the tasks and gathering data traces of their execution by users. Their findings showed that users are biased towards procedures that are (1) well-practiced and generic, and (2) composed of interactive components that are fast and incremental. Confirming the presence of the

paradox, they note that these biases tend to result in solution strategies that are slower than alternative methods.

To encourage users to improve the efficiency of their interaction, Malacria et al. [2013] proposed the use of 'skillometers', which are lightweight interactive components that promote reflection about meta-level aspects of interaction, such as task completion time or the range of commands used. A 'Skill-O-Meter' was first proposed by Linton et al. [2000], but Malacria et al. examined their potential for supporting skill development with user interfaces. Figure 10 shows an example skillometer evaluated in [Malacria et al. 2013], which shows the time taken to complete recent interface tasks, together with the time that might have been taken if an alternative scheme were used. The area labelled A in Figure 10 shows the time taken to select each of the last six commands (red if selected by pointing with the mouse; green if selected using a hotkey), as well as showing the hotkey binding for each of those commands. The area labelled B shows an aggregate performance meter, which is programmed to weight the most recent selection most highly in order to provide a clear 'reward' for hotkey use; and area C shows an estimate of the total time that might have been saved by switching to hotkeys. Comparative evaluation of a simple task with and without the skillometer suggested that the approach was successful in promoting and supporting a substantial increase in hotkey use. The 'Search Dashboard' [Bateman, Teevan and White 2012] is another example of a reflective widget (which would fit Malacria et al.'s definition of a skillometer), that was demonstrated to improve user performance in web search queries.

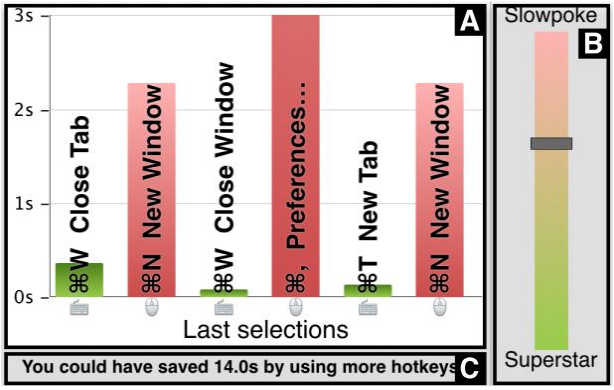


Figure 10. A skillometer widget, designed to encourage users to reflect on their own performance with a user interface and assist them in transitioning to interaction methods that offer a higher performance ceiling. Extracted from [Malacria et al. 2013].

The skillometer evaluated by Malacria et al. calculated time savings based on Keystroke Level Model [Card et al. 1983] estimates of task completion time. However, there are interesting possibilities in exaggerating the benefits in order to overcome the inertia induced by satisficing. Adar et al. [2013] provide a general discussion of the role of 'benevolent deception' in user interface design. Clearly, though, if the deception is too obvious, the user will distrust the system and stop attending to its feedback.

5.2 Performance dip after switching

The size of the performance dip that occurs after switching to a new interface will be influenced by the magnitude of the differences between the interface techniques (methods, functions, or strategies) used with the pre- and post-switch interfaces. The *rehearsal* methods used by marking menus (Section 4.2) provide an excellent example of minimising the performance dip (or eliminating it) by making the novice interface technique a rehearsal of the expert one.

The design of 'ExposeHotKey' [Malacria et al. 2013] used Kurtenbach's [1993] *rehearsal* design guideline to promote hotkey learning and use. Normally, keyboard shortcuts are only displayed to the user in response to the pointer rolling over a toolbar item or clicking on a menu. However, once the pointer has reached the item or menu, pointer-based selection is almost complete, so the user is likely to click to select. Consequently, the user's *intention* to learn a hotkey must be mediated by a different modality (the pointer), which violates the principle of rehearsal. To facilitate rehearsal of hotkey interaction, 'ExposeHotKey' displayed all keyboard shortcuts at once, overlaid on top of the normal graphical widgets, whenever a modifier key is pressed. Consequently, users could *reveal* and *rehearse* hotkeys without using the pointer. Experimental results showed that ExposeHotKey led to substantial increases in hotkey use, with 94% of selections completed using hotkeys with ExposeHotKey, compared to 50% without it.

6. DOMAIN 3: VOCABULARY EXTENSION

Many interfaces allow access to extensive functionality, often including hundreds or thousands of commands [Hsi and Potts 2000; McGrenere, Baecker and Booth 2007], yet users typically make use of only a small subset. For example, a study of the Unix operating system showed that experts used only a relatively small proportion of commands [Draper 1984] – of the ~570 commands available, Draper's study showed a peak in the participants' vocabulary size at ~45 commands. More recently, Matejka et al.'s [2009] analysis of command use in AutoCAD showed that most users had a vocabulary of ~30-40 commands, representing a small fraction of the thousands available. It is therefore likely that users often fail to make use of functions that may assist their performance, possibly due to a lack of awareness or to subconscious satisficing. For example, in a photo editing suite a user may be familiar with a tool for painting filled circles and may use it to remove red-eye from photographs, unaware that a more effective dedicated red-eye removal function is available.

The conceptual deterrents and barriers to vocabulary extension are similar to those described for *intermodal improvement* – users need to be aware of commands before using them, and they need to understand the magnitude of improvement that they will enable. Consequently, several of the methods described in the *intermodal improvement* section could also be used for increasing the user's vocabulary, such as calm notification or skillometers. Similarly, the methods used for explicit task instruction (described in Section 7), could also be used to extend the user's vocabulary. In this section, we focus on other interface methods for extending vocabulary, particularly those that use ambient suggestions and recommendations, rather than explicit instruction.

Researchers at Autodesk have conducted several studies aimed at improving users' knowledge and use of helpful interface functions. Their work is motivated by the mismatch between the large and growing number of built-in user interface commands in their software (e.g., AutoCAD's total of 750 commands in 2007) and the small command vocabulary of most users (e.g., 90% of users had a vocabulary of fewer than 90 commands) [Matejka et al. 2009; Li, Matejka, Grossman, Konstan and Fitzmaurice 2011].

Two interrelated themes of the Autodesk research on command recommendation concern understanding the best interface mechanisms for presenting recommendations to users, and devising methods to generate high quality recommendations. Their key interface goals are that the assistive content should be dynamically updated to maintain contextually relevant assistance [Ekstrand, Li, Grossman, Matejka and Fitzmaurice 2011]; that it should be continually available to the user; and that it should be presented in an ambient manner that allows access at a glance, thus minimizing impediments to access [Matejka, Grossman and Fitzmaurice 2011]. Figure 11 from Matejka, Grossman and Fitzmaurice [2013] shows

their 'Patina' interface adaptation method, which uses coloured heatmap overlays to emphasise command recommendations without impeding access to them. Related concepts of using stencils to direct the user's attention to salient interface objects were also presented by Kelleher and Pausch [2005], but for the purpose of supporting online tutorials.

In generating high quality recommendations, they focus on algorithms that produce recommendations that are both novel (rather than familiar) and useful [Matejka et al. 2009; Li et al. 2011]. They compared two forms of collaborative filtering algorithms: user-based algorithms, which generate recommendations based on the group of individuals that a particular user is most similar to, selecting those commands that are frequently used by the group of similar users; and item-based algorithms, which generate recommendations based on the similarity between the user's active command and other commands selected by the community of users. Evaluation results of their 'CommunityCommands' system suggest that an item-based collaborative filtering algorithm performs best, generating the highest number of useful recommendations as well as the fewest number of unuseful ones. While CommunityCommands uses one form of collaborative information to assist vocabulary extension, several other researchers have noted that other forms of social influence are also important. For example, Banovic et al. [2012] showed that social influence was important for encouraging tool palette customization; Peres et al. [2004] found that users were more likely to use keyboard shortcuts if they worked with others who did so; and Bateman et al. [2012] found that the queries issued to a search engine became more expert when the interface for submitting queries showed aggregate information about the user's past searches together with examples from experts' profiles.

Figure 11. Patina [Matejka et al. 2013] uses a coloured heatmap to recommend commands to users.

7. DOMAIN 4: TASK MAPPING AT THE INTERFACE

Domains 1-3 (intramodal improvement, intermodal improvement, and vocabulary extension) stem from a structural decomposition of the user interface into the performance characteristics of methods and the various methods and functions supported by the user interface (Figure 2). Domain 4, in contrast, focuses on the user's comprehension of their tasks, the strategies they use to complete them, the degree to which the interface interferes with task execution, and the interface's facilities for customization. The focus remains, however, on the components of the user interface that influence these capabilities, as described in the following four subsections.

7.1. Task training: Manuals, Online Help and Tutorials

A new user of a computer system must find ways to understand the system and the functions and methods that can be used to execute their tasks. In early HCI research, several studies observed the successes of exploratory learning in video games, leading to suggestions that designers of office systems should seek ways to intrinsically motivate users to explore the interface [Malone 1980; Carroll 1987; Shneiderman 1987]. These observations led to some important innovations, such as Carroll's successful use of 'minimal' task-centered manuals to facilitate rapid interface familiarization [Carroll, Smith-Kerker, Ford and Mazur-Rimetz 1987; Carroll 1990]. However, subsequent studies have shown that exploratory learning for

intrinsic interest is rare. Instead, users tend to focus on accomplishing real work tasks, and resort to tutorials, manuals, and trial-and-error as 'just-in-time' coping strategies [Rieman 1996]. Interfaces to online help systems have evolved substantially since Rieman's study, but more recent studies indicate that online help is still predominantly used as material of last resort [Novick and Ward 2006]. However, this situation may evolve as systems improve at extracting the user's task context (e.g., [Ekstrand et al. 2011]) and adopt ambient presentation to assure that help is continually ready to hand (e.g., [Matejka et al. 2011]).

Like online help, there has been a recent surge of research on online tutorials, with contributions from Autodesk being particularly prominent. Four emerging directions are described in the following paragraphs: gamification, community input into tutorials, leveraging transfer effects, and distribution of practice.

Gamification

Deterding et al. [2011] define gamification as "the use of design elements which are characteristic for games in non-game contexts". Typically, gamification is intended to increase the user's motivation to engage in an activity and to improve their performance at it, although other benefits such as increased collaboration and creative leadership may also occur [Reeves and Read 2009]. McGonigal [2011] describes four key characteristics of games that engender intrinsic motivation, with the characteristics reflecting those identified earlier in the review of human factors of motivation and feedback (Section 2.8): provide a goal to give the user a sense of purpose; use rules that limit how the goal can be achieved; provide feedback to show progression towards the goal; and voluntary participation, which engenders acceptance of the rules and feedback.

Researchers have contemplated or demonstrated the use of gamification in a wide range of application domains, including education [Linehan, Kirman, Lawson and Chan 2011], in-car aids [Diewald, Moller, Roalter, Stockinger and Kranz 2013], learning software applications [Dong, Dontcheva, Joseph, Karahalios, Newman and Ackerman 2012; Li, Grossman and Fitzmaurice 2012], software calibration [Flatla, Gutwin, Nacke, Bateman and Mandryk 2011] and business practice [Reeves and Read 2009]. As yet, there have been relatively few formal studies into the success of gamification methods. Two exceptions are Flatla et al. [2011] and Li et al. [2012]. Flatla et al. proposed using gamification to encourage users to undergo software calibration procedures that are sometimes necessary to optimize software to specific user characteristics, and their lab studies showed that users found gamified calibration procedures to be significantly more enjoyable, without compromising the calibration outcomes. Li et al.'s [2012] 'GamiCAD' tutorial system was intended to provide an engaging introduction to high functionality AutoCAD software. GamiCAD included various gamified components, including a 'mission console' for navigating the game, a scoreboard, speed bonuses, step-by-step instructions, and various task-specific mini-games. Formal evaluation of GamiCAD in comparison with a non-gamified tutorial showed that participants completed tasks faster and more accurately following training with GamiCAD, and that they substantially preferred using it to the traditional tutorial.

Gamification is an emerging area of research, and we anticipate future results of field testing of gamified systems applied to improving interface expertise.

Community input into tutorials

In recent work, Lafreniere et al., [2013] describe and evaluate their 'FollowUs' tutorial system, which aims to address problems stemming from a lack of connection between traditional tutorials and the associated application. They argue that this lack of connection creates several limitations in tutorial material, particularly the inability for individuals and communities to update and improve tutorial content.

FollowUs uses an 'application-in-tutorial architecture', where the application lies within the tutorial system, allowing the tutorial to capture the user's workflow while following the tutorial, and providing easy mechanisms to record alternative or improved methods that might augment the original tutorial.

A formative evaluation of FollowUs compared user performance in a series of tasks when using FollowUs and when using another condition that differed only in the exclusion of community sourced tutorials. Task completion rates were higher and frustration lower when using FollowUs. In general, recent work on crowd-sourcing and improvements in recommender algorithms are creating interesting opportunities for bringing relevant tutorial material to the user's attention, although Lafreniere et al. note that there are substantial challenges in designing interfaces that present this content to users in a manner that is helpful and timely, without being distracting.

Systems that exploit transfer effects between software versions

Ramesh, Hsu, Agrawala and Hartmann [2011] examined interface methods to help users transfer existing knowledge of one software system or version to another related system/version. Their ShowMeHow system demonstrates their concept of 'UI translation interfaces'. ShowMeHow allowed users to inspect the translation of commands between applications within the same domain, incorporating two main capabilities: the ability to click on a façade representing the known software to reveal related components in the new system; and text searches for command names in either application, based on manually constructed data structures describing terms used in both the original and new applications. An initial user study showed that users could locate unfamiliar commands in an interface much faster when using ShowMeHow, and a second study demonstrated that users could repurpose tutorials written for one application in order to learn another.

Using temporal distribution of practice in training systems

Finally, HCI researchers have used psychology findings on the effects of temporal distribution of practice (Section 2.6) to assist with learning motor aspects of interface performance. For example, studies by Zhai et al. [2002; Zhai and Kristensson 2003] examined users learning new keyboard layouts, with results suggesting effective memorization when users were trained using expanding rehearsal intervals that scheduled the reappearance of training items (such as letters or words) with increasing time intervals between exposures.

7.2. Support for advanced task strategies

Beyond initial familiarization, Bhavnani and John [2000] observe that knowledge of tasks and tools is insufficient for users to become efficient with user interfaces, and that users additionally need to attain an intermediate 'strategic' layer of knowledge. This strategic knowledge facilitates the efficient mapping from user tasks to system tools. Delaney et al.'s [1998] investigation of task performance also shows that the power law of practice improvement in a task is best analyzed with consideration to the user strategy employed by the user.

Bhavnani and John's analysis of strategic interaction reveals several common examples of efficient strategies that generalize across a variety of application domains. For example, they contrast the novice strategy of 'Sequence by Operation' (exemplified by the window drawing task shown in Figure 12a) with the more expert 'Detail, Aggregate, Manipulate' strategy (shown in Figure 12b); other strategic examples are 'Aggregate-Modify (all)-Modify (exception)' and 'Locate-Aggregate-Manipulate-Modify'. Bhavnani and John provide good examples of the expertise benefits enabled by supporting these strategies in word processors (e.g., through the use of formatting styles) and Spreadsheets (e.g., formulae that work across multiple cells).

Bhavnani and John focused on demonstrating that the strategic layer of expertise existed, and on showing that performance differences could be explained by different strategic approaches. Where design implications emerge, they revolve around ensuring that interfaces support the strategies identified (such as mechanisms to facilitate aggregation or to propagate changes across related data), and the provision of explicit strategic training.

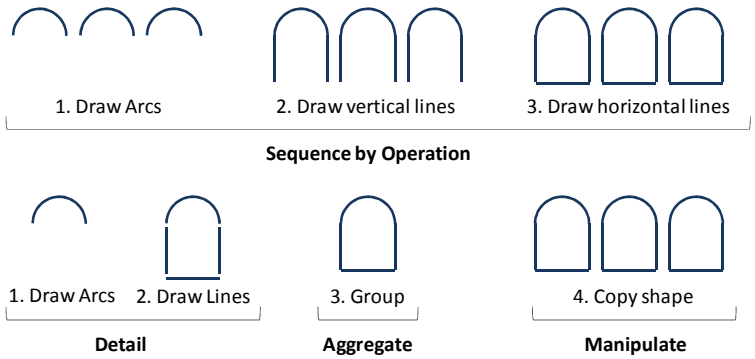


Figure 12. Two strategies for drawing three windows. Novice's "sequence by operation" strategy (top); more efficient "detail-aggregate-manipulate" strategy (bottom). Adapted from [Bhavnani and John 2000].

More recently, researchers have examined the use of automated capture, analysis and review of workflows [Grossman, Matejka and Fitzmaurice 2010; Kong, Grossman, Hartmann, Agrawala and Fitzmaurice 2012], to assist users in reflecting on their interaction strategies. For example, the Chronicle system [Grossman et al. 2010] provides a zoomable timeline that allows the user to visualize and examine workflows for completing image editing tasks, with formative evaluations suggesting that they are useful tools. Chronicle is now released as a technology preview by Autodesk, so results of field testing may appear in future research.

7.3. Task disruption and locus of control

The relevance of any system feedback to the user's current task, and the degree to which it disrupts their task execution, are important considerations for the design of interface components that aim to improve user's performance. Research by Bødker [1995] suggests that there are two types of task interruption: *breakdowns* and *focus-shifts*. Breakdowns result in severe disruption, and force the user's attention to a new activity, while focus-shifts cause only a brief attention switch and cause much less disruption. Interfaces such as Microsoft's Clippy have been shown to be quickly abandoned after a few untimely interruptions [Shroyer 2000], while less obtrusive feedback, such as Blur's 'calm notification' [Scarr et al. 2011], may go unnoticed when the user is focused on their work, resulting in a missed opportunity for improvement.

The related concept of locus of control concerns the nature of the stimulus for presenting information intended to improve the user's performance. Such information may be *pushed* by the system without any apparent triggering action from the user – for example, a 'tip of the day' message [Fischer 2001]. Alternatively, the information may be explicitly *pulled* by the user – for example, the user may intentionally dwell on a button to view its hotkey. Often, however, interface mechanisms have a more subtle interaction than explicit *push* or *pull* modalities. For example, a typeahead feature in a web browser's URL field might offer 'wikipedia.com' after the user types 'w', with the recommendation accepted by pressing <enter>. Once the user is familiar with this interaction, they might habitually use 'w <enter>' as a shortcut for the frequently visited webpage, effectively 'pulling' the feature. Alternatively, the autocorrect features used with mobile phones, which typically default to accepting the recommendation when the user presses the

space key can be viewed as 'pushing' unwanted recommendations on the user when the recommendation is incorrect.

The success of the designer's approach to locus of control is therefore influenced by many factors, including the probability that any system feedback accurately reflects the user's intention, the ease with which causal relationships between action and effect can be learned, their stability and predictability, the temporal connection between action and response, the user's degree of focus on the work environment, and the potential costs of interruption. Many of these design factors are discussed in the context of autonomous services and intelligent agents under the topic of 'mixed initiative user interfaces' [Horvitz 1999].

7.4 User interface customisation and adaptation

User interface customization facilities are an extreme form of user-centred locus of control, where the user changes the interface configuration to better match their needs. Adaptive systems lie at the opposite extreme of the locus of control spectrum, with the interface undergoing *automatic* reconfiguration based on some prediction of improved fit with user needs. Both interface customization and adaptive interfaces are substantial research areas that are beyond the scope of this review. However, some of the primary limitations of customization and adaptation are similar to those discussed above. In particular, Simon's 'satisficing' (Section 2.10) and Carroll's 'paradox of the active user' (Section 1) deter users from customizing their interfaces; and automatic adaptations are likely to cause a temporary performance dip (Section 5) due to the need for the user to identify and comprehend the interface adaptation. Those interested in interface customization and adaptable user interfaces are directed to Bunt et al. [2007] and Gajos et al. [2006].

8. DISCUSSION

This survey focused on interface techniques that are designed to assist users in gaining UI expertise. Underlying human factors that may influence the success of these techniques were briefly reviewed in Section 2. Four domains of interface performance improvement – intramodal, intermodal, vocabulary extension and task mapping – were then introduced in Section 3, and techniques within each of these domains were reviewed through Sections 4-7. Potential avenues for further work were introduced throughout the paper, with particularly promising research opportunities highlighted in the following subsections.

8.1 From lab studies to field studies

Nearly all of the interfaces described in Sections 4-7 were evaluated in controlled laboratory settings, yet several papers observe that lab studies produce more favourable findings than those conducted in the field (e.g., [Duh, Tan and Chen 2006; Fitchett, Cockburn and Gutwin 2014]). As results from human factors research show that skill acquisition can be influenced by subtle variations, it is highly desirable that the laboratory findings are validated in the field. However, developing systems that can be used in field studies of real work is extremely difficult.

The software that people use for everyday office work is often proprietary, and the lack of access to source code constrains researchers' ability to modify system behavior and monitor its use. Ideally researchers will have a relationship with the software provider, allowing direct access to source code – for example, Autodesk's 'Chronicle' research system has now been released for actual use (Section 7.2).

When source code is unavailable there are often application programming interfaces (APIs) that allow limited control of software systems running on a particular platform. Typically, these APIs form part of the accessibility framework that assists development of software for users with special needs. Several researchers

describe tools that assist with monitoring and logging interaction with unaltered software systems (e.g., [Kukreja, Stevenson and Ritter 2007; Alexander, Cockburn and Lobb 2008; Morgan, Cheng, Pike and Ritter 2013]), and these methods can also be extended to enable some control over proprietary systems, as demonstrated by the Blur and Skillometer systems (Section 5.1), which ran on Microsoft Windows and Apple Mac OSX. However, these techniques have several limitations, including access to only a subset of controls, reduced system responsiveness, and increased frequency of software crashes, which make them impractical for studies of real use.

Open source systems permit modifications enabling longitudinal analysis of expert use (e.g., Murphy et al.'s [2006] analysis of Java developers using the Eclipse IDE), but for productivity applications, the user base is often smaller than major commercial systems.

8.2 Understanding the costs and benefits of intermodal methods

Methods for intermodal improvement, such as keyboard accelerators, are already commonly used in state-of-the-art interfaces. While researchers have examined methods to minimize the intermodal performance dip (Figure 8), there are opportunities for more broadly examining the costs associated with providing more than one method for activating a function. For example, Olson and Nilsen [1988] showed that users who knew two methods for entering formulae into a spreadsheet were significantly slower than users who knew only one, and they attributed the time cost to the additional cognition required to plan and decide between methods. Similarly, Quinn et al. [2013] recently demonstrated that the decision costs associated with choosing between alternative cursor pointing paths negated the benefits of a shortcut pointing method that allowed users to wrap the cursor around display edges.

8.3 Rehearsal and guidance interfaces

As described in Section 4.2, Kurtenbach's [1993] marking menus were designed so that the novice's motor actions were a physical rehearsal of those used when expert. Recently, several researchers have generalized the principle of rehearsal in other forms of interaction, including dragged finger gestures [Bau and Mackay 2008] and hotkeys [Malacria et al. 2013]. It seems likely that the principles of rehearsal can be deployed in other forms of interaction to facilitate expertise development.

For example, FastTap [Gutwin, Cockburn, Scarr and Malacria 2014] is a recently proposed interactive technique for improving command selections on tablets – when novice, the user's selections are visually guided by the display of a spatially stable grid of menu items, but once expert multiple selections can be made by chorded taps on the known locations without the need to wait for visual guidance.

As well as devising new interface techniques based on the principle of rehearsal, there are extensive research opportunities in better understanding how interfaces can best support users through different forms of guidance. Anderson and Bischof's [2013] study (Section 4.2) provides a recent example, demonstrating that progressive removal of guidance has learning advantages over its continual availability.

8.4 Deliberate practice, its timing, content and acceptability

Section 2.3 reviewed Ericsson's research, emphasizing the role of deliberate practice in the development of elite levels of skill. Section 2.4 then reviewed the connection between the depth of mental engagement in activities and their resultant memorization. Section 2.6 also reviewed how the distribution of practice sessions influences learning.

There are relatively few examples of user interfaces that explicitly draw on these factors to promote skill development with the interface; consequently, there are many

opportunities for further work. These include the following: work on algorithms for identifying potentially beneficial methods, functions, or strategies that the user appears to be unaware of, as well as determining appropriate times and methods for their presentation – for example, Li et al.'s [2011] collaborative filtering algorithms, Section 6); user interface techniques for presenting engaging practice sessions to users – for example, gamification (Section 7.1); and research on how to optimize the level of effort required across various tasks and skill levels (e.g., an adaptive form of the frost-brushing interface described in Section 4.2, Figure 5).

8.5 Speed/accuracy/quality tradeoffs across expertise

Most of this review has focused on task performance time, but accuracy and the cost of errors are often critical to successful interaction, and they can vary substantially between interaction techniques. For example, a missed pointer-based click on Microsoft Word's 'bold' icon is likely to be quickly identified by the absence of icon highlighting, and correcting the error will be fast because the cursor is already near the target. In contrast, a user who mistakenly believes the hotkey for 'bold' is <Ctrl-L> will need to infer their error from the change of display state, and then undo the error, home their hands to the mouse, point to the 'bold' icon, dwell to view the tooltip, memorise the hotkey, home their hands back to the keyboard, and activate the hotkey. Research is needed to better understand the role of accuracy and error correction in expertise development. Related research questions arise when considering the quality of the work product rather than outright task time (e.g., the quality of a design for a designer working with a CAD package).

8.6 Interface methods for notification, and their underlying algorithms

Commercial and research user interfaces have made substantial improvements to interaction through increased use of 'mixed initiative' interactions [Bunt et al. 2007], in which the system offers non-intrusive notifications that aim to assist the task as a result of monitoring the user actions. Simple examples include 'type-ahead' features, such as presenting 'cnn.com' for immediate selection after the user types 'cn' into a URL field (Section 4.2). Four directions for subsequent research on mixed initiative interactions are: 1. examining how to use new sources of information for generating recommendations, such as crowd-sourcing help recommendations [Chilana, Ko, Wobbrock and Grossman 2013]; 2. improved algorithms for generating recommendations, such as AccessRank [Fitchett and Cockburn 2012]; 3. clarifying the relationship between post-action feedback and during-action feedforward (e.g, Vermeulen, Luyten, Hoven and Coninx [2013]); and 4. new mixed initiative interactions (e.g., Octopocus, Section 4.2).

8.7 Generalisability of previous results

Human factors literature, summarized in Section 2, shows that many factors influence skill acquisition. Interaction with computing systems introduces new and important factors that have not been extensively studied. It is therefore important that research findings are validated in varied settings, with diverse user groups and over prolonged periods. Factors that might influence results include different input and output configurations, and how these configurations interact with task factors.

For example, it is common for users in high workload situations (such as air traffic control, stock trading, or emergency dispatch) to use multiple displays to monitor concurrent activities. While there has been research on how highly trained experts coordinate their activities in such settings [Hornof, Zhang and Halverson 2010] there is scope for research into skill acquisition in such settings. Other factors that might be examined include *user factors* (culture, stress, age, etc.), *task factors* (e.g., high cognitive demand versus low, workload, task pace), *input device* (e.g., touchscreen,

keyboard, mouse, trackpad), and *output configuration* (e.g., the size and location of data on single or multiple displays, and the effect of output modalities such as sound and haptic feedback).

9. CONCLUSIONS

Graphical user interfaces have become the primary medium through which office work is conducted. A primary factor contributing to their success is the ease with which they can be learned – newcomers can quickly attain sufficient competence to adequately complete their work. While ease of learning is highly desirable, it is common for office workers to continue using substantially similar interfaces for months, years, and decades, so optimizing designs for initial learnability rather than long term efficiency can be counterproductive. Current interface guidelines attempt to account for novice learning *and* expertise by advocating for shortcut facilities, but this strategy is known to fail as few users make the transition to expert methods – instead, users maintain the adequate but inefficient strategies they learned first.

There has been a recent surge of research interest in resolving these problems of graphical user interfaces tending to trap users in a beginner mode of operation. While some of these research projects are founded on established findings from the human factors literature, others are not. This paper provided a review of key findings from the human factors literature that are relevant to supporting transitions from novice to expert performance with user interfaces, as well as summarizing the state of the art in interface research supporting the transition. We provided directions for further work and look forward to a new generation of user interfaces that facilitate expertise as well as ease of use.

REFERENCES

ACKERMAN, P. L. 2007. New developments in understanding skilled performance. Current Directions in Psychological Science **16**(5): 235-239.

ADAR, E., TAN, D. S. and TEEVAN, J. 2013. Benevolent deception in human computer interaction. Proceedings of the 2013 ACM annual conference on Human factors in computing systems, Paris, France, 1863-1872. ACM.

ALEXANDER, J. 2009. Understanding and improving navigation within electronic documents. Computer Science and Software Engineering. Christchurch, University of Canterbury. **PhD**.

ALEXANDER, J., COCKBURN, A. and LOBB, R. 2008. Appmonitor: A tool for recording user actions in unmodified windows applications. Behavior Research Methods **40**(2): 413-421.

ANDERSON, F. and BISCHOF, W. F. 2013. Learning and performance with gesture guides. Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, Paris, France, 1109-1118. ACM.

ANDERSON, J. 1995. Learning and memory. New York, John Wiley.

ANDERSON, J. 2005. Cognitive psychology and its implications. New York, Worth Publishers.

APPERT, C. and BAU, O. 2010. Scale detection for a priori gesture recognition. Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, Atlanta, Georgia, USA, 879-882. ACM.

ARMSTRONG, T. 1970. Training for the production of memorized movement patterns. Ann Arbor, MI, University of Michigan, Department of Psychology.

BADDELEY, A. and LONGMAN, D. 1978. The influence of length and frequency of training session on the rate of learning to type. Ergonomics **21**: 627-635.

BADDELEY, A. D. 1999. Essentials of human memory. Hove, England, Psychology Press.

BANOVIC, N., CHEVALIER, F., GROSSMAN, T. and FITZMAURICE, G. 2012. Triggering triggers and burying barriers to customizing software. Proceedings of the 2012 ACM annual conference on Human Factors in Computing Systems, Austin, Texas, USA, 2717-2726. ACM.

BARRETT, R., KANDOGAN, E., MAGLIO, P., HABER, E., TAKAYAMA, L. and PRABAKER, M. 2004. Field studies of computer system administrators: Analysis of system management tools and practices. Proceedings of CSCW'04: ACM conference on Computer supported cooperative work, Chicago, Illinois, USA, 388-395. ACM Press.

BATEMAN, S., TEEVAN, J. and WHITE, R. W. 2012. The search dashboard: How reflection and comparison impact search behavior. Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, Austin, Texas, USA, 1785-1794. ACM.
BAU, O. and MACKAY, W. E. 2008. Octopocus: A dynamic guide for learning gesture-based command sets. Proceedings of the 21st annual ACM symposium on User interface software and technology, Monterey, CA, USA, 37-46. ACM.
BENNETT, M., MCCARTHY, K., O'MODHRAIN, S. and SMYTH, B. 2011. Simpleflow: Enhancing gestural interaction with gesture prediction, abbreviation and autocompletion. Proceedings of the 13th IFIP TC 13 international conference on Human-computer interaction - Volume Part I, Lisbon, Portugal, 591-608. Springer-Verlag.
BHAVNANI, S. and JOHN, B. 2000. The strategic use of complex computer systems. Human Computer Interaction **15**: 107-137.
BODKER, S. 1995. Applying activity theory to video analysis: How to make sense of video data in human-computer interaction. Context and consciousness, Massachusetts Institute of Technology**:** 147-174.
BOURNE, L. and ARCHER, E. 1956. Time continuously on target as a function of distribution of practice. Journal of Experimental Psychology **51**: 25-33.
BULLOCK, W. A. and GILLILAND, K. 1993. Eysenck's arousal theory of introversion€xtraversion: A converging measures investigation. Journal of Personality and Social Psychology **64**(1): 113-123.
BUNT, A., CONATI, C. and MCGRENERE, J. 2007. Supporting interface customization using a mixed-initiative approach. Proceedings of IUI '07: Conference on Intelligent User Interfaces, Honolulu, Hawaii, 92-101. ACM.
CARD, S. K., MORAN, T. P. and NEWELL, A. 1983. The psychology of human-computer interaction, Lawrence Erlbaum Associates.
CARROLL, J. 1987. The adventure of getting to know a computer. Readings in human-computer interaction: A multidisciplinary approach. BAECKER, R. and BUXTON, W., Morgan Kaufmann Publishers Inc. **:** 639-648.
CARROLL, J. 1990. The nurnberg funnel: Designing minimalist instruction for practical computer skill. Cambridge, MA, MIT Press.
CARROLL, J. and CARRITHERS, C. 1984. Training wheels in a user interface. Communications of the ACM **27**(8): 800-806.
CARROLL, J. and ROSSEN, M. 1987. Paradox of the active user. Interfacing thought: Cognitive aspects of human-computer interaction. CARROLL, J. Cambridge, MA, MIT Press**:** 80-111.
CARROLL, J., SMITH-KERKER, P., FORD, J. and MAZUR-RIMETZ, S. 1987. The minimal manual. Human Computer Interaction **3**(2): 123-153.
CEPEDA, N., PASHLER, H., VUL, E., WIXTED, J. and ROHRER, D. 2006. Distributed practice in verbal recall tasks: A review and quantitative synthesis. Psychological Bulletin **132**(3): 354-380.
CHILANA, P. K., KO, A. J., WOBBROCK, J. O. and GROSSMAN, T. 2013. A multi-site field study of crowdsourced contextual help: Usage and perspectives of end users and software teams. Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, Paris, France, 217-226. ACM.
COCKBURN, A. and GUTWIN, C. 2009. A predictive model of human performance with scrolling and hierarchical lists. Human-Computer Interaction **24**(3): 273-314.
COCKBURN, A., GUTWIN, C. and GREENBERG, S. 2007. A predictive model of menu performance. Proceedings of CHI'07: ACM Conference on Human Factors in Computing Systems, San Jose, CA, 627-636. ACM Press.
COCKBURN, A., KRISTENSSON, P., ALEXANDER, J. and ZHAI, S. 2007. Hard lessons: Effort-inducing interfaces benefit spatial learning. Proceedings of CHI'07: ACM Conference on Human Factors in Computing Systems, San Jose, CA, 1571-1580. ACM Press.
COCKBURN, A. and MCKENZIE, B. 2002. Evaluating the effectiveness of spatial memory in 2d and 3d physical and virtual environments. Proceedings of CHI'2002 Conference on Human Factors in Computing Systems, Minneapolis, Minnesota, 203-210. ACM Press.
CRAIK, F. and LOCKHART, R. 1972. Levels of processing: A framework for memory research. Journal of Verbal Learning and Verbal Behavior **11**: 671-684.

CRAIK, F. and TULVING, E. 1975. Depth of processing and the retention of words in episodic memory. Journal of Experimental Psychology: General **103**(3): 268-294.
CROSSMAN, E. 1959. A theory of acquisition of speed-skill. Ergonomics **2**(2): 153-166.
CZERWINSKI, M., HORVITZ, E. and CUTTRELL, E. 2001. Subjective duration assessment: An impicit probe for software usability. Proceedings of IHM-HCI, Lille, France, 167-170.
DAHLIN, B. and WATKINS, D. 2000. The role of repetition in the processes of memorising and understanding: A comparison of the views of german and chinese secondary school students in hong kong. British Journal of Educational Psychology **70**(1): 65-84.
DELANEY, P. F., REDER, L. M., STASZEWSKI, J. J. and RITTER, F. E. 1998. The strategy-specific nature of improvement: The power law applies by strategy within task. Psychological Science **9**(1): 1-7.
DETERDING, S., DIXON, D., KHALED, R. and NACKE, L. 2011. From game design elements to gamefulness: Defining "gamification". Proceedings of the 15th International Academic MindTrek Conference: Envisioning Future Media Environments, Tampere, Finland, 9-15. ACM.
DIEWALD, S., MOLLER, A., ROALTER, L., STOCKINGER, T. and KRANZ, M. 2013. Gameful design in the automotive domain: Review, outlook and challenges. Proceedings of the 5th International Conference on Automotive User Interfaces and Interactive Vehicular Applications, Eindhoven, Netherlands, 262-265. ACM.
DIX, A., FINLAY, J., ABOWD, G. and BEALE, R. 1993. Human-computer interaction. Harlow, England, Prentice Hall.
DOANE, S., PELLEGRINO, J. and KLATZKY, R. 1990. Expertise in a computer operating system: Conceptualization and performance. Human Computer Interaction **5**: 267-304.
DONG, T., DONTCHEVA, M., JOSEPH, D., KARAHALIOS, K., NEWMAN, M. and ACKERMAN, M. 2012. Discovery-based games for learning software. Proceedings of the 2012 ACM annual conference on Human Factors in Computing Systems, Austin, Texas, USA, 2083-2086. ACM.
DRAPER, S. 1984. The nature of expertise in unix. INTERACT 84: 1st IFIP International Conference on Human-Computer Interaction, London, UK, 465-471. North-Holland.
DUH, H. B.-L., TAN, G. C. B. and CHEN, V. H.-H. 2006. Usability evaluation for mobile device: A comparison of laboratory and field tests. Proceedings of the 8th conference on Human-computer interaction with mobile devices and services, Helsinki, Finland, 181-186. ACM.
DYCK, J., PIENELLE, D., BROWN, B. and GUTWIN, C. 2003. Learning from games: Hci innovations in entertainment software. Proceedings of Graphics Interface, Halifax, 237-246. AK Peters.
EKSTRAND, M., LI, W., GROSSMAN, T., MATEJKA, J. and FITZMAURICE, G. 2011. Searching for software learning resources using application context. Proceedings of the 24th annual ACM symposium on User interface software and technology, Santa Barbara, California, USA, 195-204. ACM.
ENGSTRÖM, J., JOHANSSON, E. and ÖSTLUND, J. 2005. Effects of visual and cognitive load in real and simulated motorway driving. Transportation Research Part F: Traffic Psychology and Behaviour **8**(2): 97-120.
ERICSSON, K. 2004. Deliberate practice and the acquisition and maintenance of expert performance in medicine and related domains. Academic Medicine **79**(10): S1-S12.
ERICSSON, K., PRIETULA, M. and COKELY, E. 2007. The making of an expert. Harvard Business Review **July-August**: 1-7.
FINDLATER, L., MOFFATT, K., MCGRENERE, J. and DAWSON, J. 2009. Ephemeral adaptation: The use of gradual onset to improve menu selection performance. Proceedings CHI'09 Conference on Human Factors in Computing Systems, Boston, MA, 1655-1664. ACM Press.
FISCHER, G. 2001. User modeling in human-computer interaction. User Modeling and User-Adapted Interaction **11**(1/2): 65--86.
FITCHETT, S. and COCKBURN, A. 2012. Accessrank: Predicting what users will do next. Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, Austin, Texas, USA, 2239-2242. ACM.
FITCHETT, S., COCKBURN, A. and GUTWIN, C. 2014. Finder highlights: Field evaluation and design of an augmented file browser. Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, Paris, France, In Press. ACM.
FITTS, P. 1964. Perceptual-motor skills learning. Categories of human learning. MELTON, A. New York, Academic Press**:** 243-285.

FITTS, P. M. and POSNER, M. I. 1967. Human performance. Belmont, CA, Brookes/Cole.
FLATLA, D. R., GUTWIN, C., NACKE, L. E., BATEMAN, S. and MANDRYK, R. L. 2011. Calibration games: Making calibration tasks enjoyable by adding motivating game elements. Proceedings of the 24th annual ACM symposium on User interface software and technology, Santa Barbara, California, USA, 403-412. ACM.
FREEMAN, D., BENKO, H., MORRIS, M. R. and WIGDOR, D. 2009. Shadowguides: Visualizations for in-situ learning of multi-touch and whole-hand gestures. Proceedings of the ACM International Conference on Interactive Tabletops and Surfaces, Banff, Alberta, Canada, 165-172. ACM.
FU, W.-T. and GRAY, W. D. 2004. Resolving the paradox of the active user: Stable suboptimal performance in interactive tasks. Cognitive Science **28**(6): 901-935.
GAJOS, K., CZERWINSKI, M., TAN, D. and WELD, D. 2006. Exploring the design space for adaptive graphical user interfaces. Proceedings of AVI'06: ACM Conference on Advanced Visual Interfaces, Venice, Italy, 201-208. ACM Press.
GENTILE, A. 1972. A working model of skill acquistion. Quest **17**: 3-23.
GRAY, W. D., SIMS, C., FU, W.-T. and SCHOELLES, M. 2006. The soft constraints hypothesis: A rational analysis approach to resource allocation for interactive behavior. Psychological Review **113**: 461-482.
GREENBERG, S. and WITTEN, I. 1993. Supporting command reuse: Empirical foundations and principles. International Journal of Man-Machine Studies **39**: 353-390.
GROSSMAN, T., DRAGICEVIC, P. and BALAKRISHNAN, R. 2007. Strategies for accelearating on-line learning of hotkeys. Proceedings of CHI'07: ACM Conference on Human Factors in Computing Systems, San Jose, California, 1591-1600. ACM Press.
GROSSMAN, T., MATEJKA, J. and FITZMAURICE, G. 2010. Chronicle: Capture, exploration, and playback of document workflow histories. Proceedings of the 23nd annual ACM symposium on User interface software and technology, New York, New York, USA, 143-152. ACM.
GUTWIN, C., COCKBURN, A., SCARR, J. and MALACRIA, S. 2014. Faster command selection on tablets with fasttap. Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, Paris, France, In Press. ACM.
HASEGAWA, M., CARPENTER, P. A. and JUST, M. A. 2002. An fmri study of bilingual sentence comprehension and workload. NeuroImage **15**(3): 647-660.
HEATHCOTE, A., BROWN, S. and MEWHORT, D. 2000. The power law repealed: The case for an exponential law of practice. Psychonomic Bulletin and Review **7**(2): 185-207.
HENDY, J., BOOTH, K. and MCGRENERE, J. 2010. Graphically enhanced keyboard accelerators for guis. Proceedings of Graphics Interface, Ottawa, Ontario, 3-10
HEWSON, M. and LITTLE, M. 2001. Giving feedback in medical education. Journal of General Internal Medicine **13**(2): 111-116.
HORNOF, A., J. , ZHANG, Y. and HALVERSON, T. 2010. Knowing where and when to look in a time-critical multimodal dual task. Proceedings of the 28th international conference on Human factors in computing systems, Atlanta, Georgia, USA, 2103-2112. ACM.
HORNOF, A. J. and KIERAS, D. E. 1997. Cognitive modeling reveals menu search is both random and systematic. Proceedings of CHI'97: ACM Conference on Human Factors in Computing Systems, Atlanta, Georgia, 107-114.
HORVITZ, E. 1999. Principles of mixed-initiative user interfaces. Proceedings of the SIGCHI conference on Human Factors in Computing Systems, Pittsburgh, Pennsylvania, USA, 159-166. ACM.
HSI, I. and POTTS, C. 2000. Studying the evolution and enhancement of software features. Proceedings of the International Conference on Software Maintenance (ICSM'00), 143. IEEE Computer Society.
HYÖNÄ, J., TOMMOLA, J. and ALAJA, A.-M. 1995. Pupil dilation as a measure of processing load in simultaneous interpretation and other language tasks. The Quarterly Journal of Experimental Psychology Section A **48**(3): 598-612.
KAHNEMAN, D. 1973. Attention and effort. New-Jersey, Prentice-Hall Inc.
KARAT, J., KARAT, C. and UKELSON, J. 2000. Affordances, motivation and the design of user interfaces. Communications of the ACM **43**(8): 49-51.

KELLEHER, C. and PAUSCH, R. 2005. Stencils-based tutorials: Design and evaluation. Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, Portland, Oregon, USA, 541-550. ACM.
KIM, J. and RITTER, F. 2013. Learning, forgetting, and relearning for keystroke- and mouse-driven tasks: Relearning is important. Human-Computer Interaction: In press.
KONG, N., GROSSMAN, T., HARTMANN, B., AGRAWALA, M. and FITZMAURICE, G. 2012. Delta: A tool for representing and comparing workflows. Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, Austin, Texas, USA, 1027-1036. ACM.
KRISLER, B. and ALTERMAN, R. 2008. Training towards mastery: Overcoming the active user paradox. Proceedings of the 5th Nordic conference on Human-computer interaction: building bridges, Lund, Sweden, 239-248. ACM.
KRISTENSSON, P. O. and DENBY, L. C. 2011. Continuous recognition and visualization of pen strokes and touch-screen gestures. Proceedings of the Eighth Eurographics Symposium on Sketch-Based Interfaces and Modeling, Vancouver, British Columbia, Canada, 95-102. ACM.
KUKREJA, U., STEVENSON, W. and RITTER, F. 2007. Rui: Recording user input from interfaces under windows and mac os x. Behavior Research Methods, Instruments, and Computers **38**(4): 656-659.
KURTENBACH, G. 1993. The design and evaluation of marking menus. Department of Computer Science. Toronto, University of Toronto. **Ph.D.**
KURTENBACH, G. and BUXTON, B. 1993. The limits of expert performance using hierarchic marking menus. Proceedings of InterCHI'93, 482-487.
KURTENBACH, G. and BUXTON, W. 1994. User learning and performance with marking menus. Proceedings of CHI'94 Conference on Human Factors in Computing Systems, Boston, 258-264. ACM.
KURTENBACH, G., MORAN, T. and BUXTON, W. 1994. Contextual animation of gestural commands. Computer Graphics Forum **13**(5): 305-314.
KURTENBACH, G., SELLEN, A. and BUXTON, W. 1993. An empirical evaluation of some articulatory and cognitive aspects of marking menus. Human-Computer Interaction **8**(1): 1-23.
KYLLO, L. B. and LANDERS, D. M. 1995. Goal setting in sport and exercise: A research synthesis to resolve the controversy. Journal of Sport & Exercise Psychology **17**(2): 117-137.
LAFRENIERE, B., GROSSMAN, T. and FITZMAURICE, G. 2013. Community enhanced tutorials: Improving tutorials with multiple demonstrations. Proceedings of CHI'13: ACM Conference on Human Factors in Computing Systems, Paris, France, 1779-1788. ACM.
LANE, D. M., NAPIER, H. A., PERES, S. C. and SANDOR, A. 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): 133-144.
LI, W., GROSSMAN, T. and FITZMAURICE, G. 2012. Gamicad: A gamified tutorial system for first time autocad users. Proceedings of the 25th annual ACM symposium on User interface software and technology, Cambridge, Massachusetts, USA, 103-112. ACM.
LI, W., MATEJKA, J., GROSSMAN, T., KONSTAN, J. A. and FITZMAURICE, G. 2011. Design and evaluation of a command recommendation system for software applications. ACM Trans. Comput.-Hum. Interact. **18**(2): 1-35.
LINEHAN, C., KIRMAN, B., LAWSON, S. and CHAN, G. 2011. Practical, appropriate, empirically-validated guidelines for designing educational games. Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, Vancouver, BC, Canada, 1979-1988. ACM.
LINTON, F., JOY, D., SCHAEFER, H. and CHARRON, A. 2000. Owl: A recommender system for organization-wide learning. Educational Technology & Society **3**(1).
LOCKE, E. and LATHAM, G. 2006. New directions in goal-setting theory. Current Directions in Psychological Science **15**: 265-268.
MALACRIA, S., BAILLY, G., HARRISON, J., COCKBURN, A. and GUTWIN, C. 2013. Promoting hotkey use through rehearsal with exposehk. Proceedings of CHI'13: ACM Conference on Human Factors in Computing Systems, Paris, France, 573-582. ACM.
MALACRIA, S., SCARR, J., COCKBURN, A., GUTWIN, C. and GROSSMAN, T. 2013. Skillometers: Reflective widgets that motivate and help users to improve performance. Proceedings of the ACM Symposium on User Interface Software and Technology (UIST '13), St. Andrews, Scotland, 321-330. ACM.

MALONE, T. 1980. What makes things fun to learn? A study of intrinsically motivating computer games. Department of Psychology. Palo Alto, CA, Stanford University.
MATEJKA, J., GROSSMAN, T. and FITZMAURICE, G. 2011. Ambient help. Proceedings of CHI'11: ACM Conference on Human Factors in Computing Systems, Vancouver, Canada, 2751-2760. ACM Press.
MATEJKA, J., GROSSMAN, T. and FITZMAURICE, G. 2013. Patina: Dynamic heatmaps for visualizing application usage. Proceedings of CHI'13: ACM Conference on Human Factors in Computing Systems, Paris, France, 3227-3236. ACM.
MATEJKA, J., LI, W., GROSSMAN, T. and FITZMAURICE, G. 2009. Communitycommands: Command recommendations for software applications. ACM Symposium on User Interface Software and Technology, 193-202. ACM.
MCGONIGAL, J. 2011. Reality is broken: Why games make use better and how they can change the world, Penguin Group.
MCGRENERE, J., BAECKER, R. M. and BOOTH, K. S. 2007. A field evaluation of an adaptable two-interface design for feature-rich software. ACM Trans. Comput.-Hum. Interact. **14**(1): 3.
MORGAN, J. H., CHENG, C.-Y., PIKE, C. and RITTER, F. E. 2013. A design, tests and considerations for improving keystroke and mouse loggers. Interacting with Computers **25**(3): 242-258.
MURPHY, G. C., KERSTEN, M. and FINDLATER, L. 2006. How are java software developers using the eclipse ide? IEEE Softw. **23**(4): 76-83.
NAVEH-BENJAMIN, M. 1987. Recognition memory of spatial location information: Another failure to support automaticity. Memory and Cognition **16**: 437-445.
NEWELL, A. and ROSENBLOOM, P. 1981. Mechanisms of skill acquisition and the law of practice. Cognitive skills and their acquisition. ANDERSON, J. Hillsdale, NJ, Erlbaum**:** 1-55.
NEWELL, A. and ROSENBLOOM, P. S. 1981. Mechanisms of skill acquisition and the law of practice. Cognitive skills and their acquisition. ANDERSON, J. Hillsdale, NJ, Erlbaum**:** 1-55.
NIELSEN, J. 1993. Usability engineering. San Francisco, Morgan Kaufmann.
NILSEN, E., JONG, H., OLSON, J. S., BIOLSI, K., RUETER, H. and MUTTER, S. 1993. The growth of software skill: A longitudinal look at learning and performance. Proceedings of the INTERACT '93 and CHI '93 Conference on Human Factors in Computing Systems, Amsterdam, The Netherlands, 149-156. ACM.
NORMAN, D. 1983. Design principles for human-computer interaction. Proceedings of chi'83 conference on human factors in computing systems**:** 1--20.
NORMAN, D. 1983. Design principles for human-computer interfaces. Proceedings of'chi 83**:** 1--10.
NOVICK, D. G. and WARD, K. 2006. Why don't people read the manual? Proceedings of the 24th annual ACM international conference on Design of communication, Myrtle Beach, SC, USA, 11-18. ACM.
ODELL, D., L. , DAVIS, R., C., SMITH, A. and WRIGHT, P., K. 2004. Toolglasses, marking menus, and hotkeys: A comparison of one and two-handed command selection techniques. Proceedings of Graphics Interface 2004, London, Ontario, Canada, 17-24. Canadian Human-Computer Communications Society.
OLSON, J. R. and NILSEN, E. L. 1988. Analysis of cognition involved in spreadsheet software interaction. Human Computer Interaction **3**(4): 309-350.
PERES, S., TAMBORELLO, F., FLEETWOOD, M., CHUNG, P. and PAIGE-SMITH, D. 2004. Keyboard shortcut usage: The roles of social factors and computer experience. Proceedings of the Human Factors and Ergonomics Society Annual Meeting, 803-807.
PERKINS-CECCATO, N., PASSMORE, S. and LEE, T. 2003. Effects of focus of attention depend on golfers' skill. Journal of Sports Sciences **21**: 593-600.
PLIMMER, B., CROSSAN, A., BREWSTER, S. A. and BLAGOJEVIC, R. 2008. Multimodal collaborative handwriting training for visually-impaired people. Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, Florence, Italy, 393-402. ACM.
QUINN, P., COCKBURN, A. and DELAMARCHE, J. 2013. Examining the costs of multiple trajectory pointing techniques. International Journal of Human-Computer Studies **71**(4): 492-509.
RAMESH, V., HSU, C., AGRAWALA, M. and HARTMANN, B. 2011. Showmehow: Translating user interface instructions between applications. Proceedings of the 24th annual ACM symposium on User interface software and technology, Santa Barbara, California, USA, 127-134. ACM.

RANDALL, N. and PEDERSEN, I. 1998. Who exactly is trying to help us? The ethos of help systems in popular computer applications. Proceedings of the 16th annual international conference on Computer documentation, Quebec, Quebec, Canada, 63-69. ACM.
REEVES, B. and READ, J. 2009. Total engagement: Using games and virtual worlds to change the way people work and businesses compete, Harvard Business Press.
REVELLE, W., HUMPHREYS, M., SIMON, L. and GILLILAND, K. 1980. The interactive effect of personality, time of day, and caffine: A test of the arousal model. Journal of Experimental Psychology **109**(1): 1-31.
RHIENMORA, P., GAJANANAN, K., HADDAWY, P., SUEBNUKARN, S., DAILEY, M. N., SUPATARATARN, E. and SHRESTHA, P. 2010. Haptic augmented reality dental trainer with automatic performance assessment. Proceedings of the 15th international conference on Intelligent user interfaces, Hong Kong, China, 425-426. ACM.
RIEMAN, J. 1996. A field study of exploratory learning strategies. ACM Trans. Comput.-Hum. Interact. **3**(3): 189-218.
ROSSON, M. B. 1983. Patterns of experience in text editing. Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, Boston, Massachusetts, USA, 171-175. ACM.
ROTHSTEIN, A. and ARNOLD, R. 1976. Bridging the gap: Application of research on videotape feedback and bowling. Motor Skills:Theory Into Practice **1**: 35-62.
SALTHOUSE, T. 1985. Anticipatory processes in transcription typing. Journal of Applied Psychology **70**: 264-271.
SCARR, J., COCKBURN, A., GUTWIN, C. and BUNT, A. 2012. Improving command selection with commandmaps. Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, Austin, Texas, USA, 257-266. ACM.
SCARR, J., COCKBURN, A., GUTWIN, C. and MALACRIA, S. 2013. Testing the robustness and performance of spatially consistent interfaces. Proceedings of the 2013 ACM annual conference on Human factors in computing systems, Paris, France, 3139-3148. ACM.
SCARR, J., COCKBURN, A., GUTWIN, C. and QUINN, P. 2011. Dips and ceilings: Understanding and supporting transitions to expertise in user interfaces. Proceedings of CHI'11: ACM Conference on Human Factors in Computing Systems, Vancouver, Canada, 2741-2750. ACM Press.
SCHMIDT, R. 1991. Frequent augmented feedback can degrade learning: Evidence and interpretations. Tutorials in motor neuroscience. REQUIN, J. and STELMACH, G.**:** 59-75.
SCHMIDT, R. and BJORK, R. 1992. The conceptualizations of practice: Common principles in three paradigms suggest new concepts for training. Psychological Science **3**(4): 207-217.
SCHMIDT, R. and LEE, T. 2011. Motor control and learning: A behavioral emphasis. New York, Human Kinetics.
SCHNEIDER, W. and SHIFFRIN, R. 1977. Controlled and automatic human information processing: I. Detection, search, and attention. Psychological Review **84**(1): 1-66.
SHNEIDERMAN, B. 1987. Direct manipulation: A step beyond programming languages (excerpt). Readings in human-computer interaction: A multidisciplinary approach. BAECKER, R. and BUXTON, W., Morgan-Kauffman**:** 461-467.
SHNEIDERMAN, B. 1992. Designing the user interface, Addison-Wesley.
SHNEIDERMAN, B. 2003. Promoting universal usability with multi-layer interface design. Proceedings of the conference on Universal Usability, Vancouver, Canada, 1-8. ACM Press.
SHROYER, R. 2000. Actual readers versus implied readers: Role conflicts in office 97. Journal of the Society for Technical Communication **47**(2): 238-40.
SIMON, H. 1959. Theories of decision-making in economics and behavioral science. The American Economic Review **49**(3): 252-283.
SNODDY, G. 1926. Learning and stability: A psychopysical analysis of a case of motor learning with clinical applications. Journal of Applied Psychology **10**: 1-36.
TAK, S., WESTENDORP, P. and VAN ROOIJ, I. 2013. Satisficing and the use of keyboard shortcuts: Being good enough is enough? Interacting with Computers **16**(1): 16.
TAN, D. S., STEFANUCCI, J. K., PROFFITT, D. R. and PAUSCH, R. 2001. The infocockpit: Providing location and place to aid human memory. Proceedings of the 2001 workshop on Perceptive user interfaces, Orlando, Florida, 1-4. ACM.

TAUSCHER, L. and GREENBERG, S. 1997. How people revisit web pages: Empirical findings and implications for the design of history systems. International Journal of Human Computer Studies, Special Issue on World Wide Web Usability **47**(1): 97-138.
THOMAS, G. 2013. Education: A very short introduction, Oxford University Press.
TUBBS, M. 1986. Goal setting: A meta-analystic examination of empirical evidence. Journal of Applied Psychology **71**(3): 474-483.
VAN ASSELEN, M., FRITSCHY, E. and POSTMA, A. 2005. The influence of intentional and incidental learning on acquiring spatial knowledge during navigation. Psychological Research **70**(2): 151-156.
VERMEULEN, J., LUYTEN, K., HOVEN, E. V. D. and CONINX, K. 2013. Crossing the bridge over norman's gulf of execution: Revealing feedforward's true identity. Proceedings of the 2013 ACM annual conference on Human factors in computing systems, Paris, France, 1931-1940. ACM.
WALLACE, S. A. and HAGLER, R. W. 1979. Knowledge of performance and the learning of a closed motor skill. Research Quarterly. American Alliance for Health, Physical Education, Recreation and Dance **50**(2): 265-271.
WHITESIDE, J., JONES, S., LEVY, P. and WIXON, D. 1985. User performance with command, menu, and iconic interfaces. Proceedings of CHI '85: ACM Conference on Human Factors in Computing Systems, 185-191. ACM Press.
WOLFE, J. 1998. What can 1 million trials tell us about visual search. Psychological Science **9**(1): 33-97.
WULF, G. and SHEA, C. 2002. Principles derived from the study of simple skills do not generalize to complex skill learning. Psychonomic Bulletin and Review **9**(2): 185-211.
YECHIAM, E., EREV, I., YEHENE, V. and GOPHER, D. 2003. Melioration and the transition from touch-typing training to everyday use. Human Factors **45**(4): 671-684.
ZHAI, S. and KRISTENSSON, P. 2003. Shorthand writing on stylus keyboard. Proceedings of CHI'2003 Conference on Human Factors in Computing Systems, Fort Lauderdale, Florida, 97-104.
ZHAI, S., SUE, A. and ACCOT, J. 2002. Movement model, hits distribution and learning in virtual keyboarding. Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, Minneapolis, Minnesota, USA, 17-24. ACM.

Appendix D

Example of Experimental Consent Form

EXPERIMENT CONSENT FORM

We (Joey Scarr and Andy Cockburn) are carrying out an investigation into pointing to on-screen targets using different interface types. Each trial involves pointing to and clicking on a target control using one of three different interfaces. In total, your participation should take approximately thirty minutes.

This is not in any way a test of your competence with computers.

All references to participants in the investigation will be anonymous.

Thank you for your co-operation. If you have any questions regarding this investigation, please contact us.

Andy Cockburn
andy@cosc.canterbury.ac.nz

Joey Scarr
joey@cosc.canterbury.ac.nz

I consent to act as a participant in an experiment that will test pointing to on-screen targets using a variety of different interfaces. I agree to let the resulting data be used for analysis and presentation subject to the conditions below:

only Joey Scarr and Andy Cockburn will have access to my data and be able to identify me from it;

data presented or published will be stripped of my identity;

I retain the right to stop my role as a test user at any time without question, and to have my data discarded.

Name:

Signature:

Date: