ACT-R Based Models for Learning Interactive Layouts

Arindam Das

A Dissertation Submitted To
The Faculty Of Graduate Studies
In Partial Fulfillment Of The Requirements
For The Degree Of
Doctor Of Philosophy

Graduate Program In Computer Science And
Engineering
York University
Toronto, Ontario

June 2014

Abstract

This dissertation presents research on learning of interactive layouts. I develop two models based on a theory of cognition known as ACT-R (Adaptive Control of Thought–Rational). I validate them against experimental data collected by other researchers.

The *first* model is a simulation model that emulates the transition from novice to expert level in text entry. The model transcribes the presented English letters on a traditional phone keypad. It predicts the non-movement time to copy a pre-cued letter. It explains the visual exploration strategy that a user may employ in the novice to expert continuum. The *second* model is a closed-form model that accounts for the combined effect of *practice*, *decay*, *proactive interference* and *mental effort* on task completion time while practicing target acquisition on an interactive layout. The model can quantitatively compare a set of layouts in terms of the *mental effort* expended to learn them.

My *first model* provides insight into how much practice is needed by a learner to progress from novice to expert level for an interactive layout. My *second model* provides insight into how effortful is it to learn a layout relative to other layouts.

Acknowledgements

I am thankful to my supervisor, Professor Dr. Wolfgang Stuerzlinger, for his guidance and financial support. I sincerely thank my committee members, Dr. Melanie Baljko, Dr. Yves Lesperance, Dr. Jill B. Rich, and Dr. Jia Xu for their comments and suggestions. I thank all of them for spending their valuable time to critically review my work.

I am thankful to my external examiner, Professor Dr. Frank E. Ritter of Penn State University, for his invaluable comments and suggestions. His comments have helped me to improve my dissertation significantly.

I thank Seela Balkissoon, Susan Cameron, Ouma Jaipaul-Gill, and Ulya Yigit for their administrative help. I thank Dr. Ahmed Sabbir Arif, Dr. Andriy Pavlovych, and Loutfouz Zaman, for their comments and suggestions on my research work. I thank Dan Bothell of Carnegie Mellon University for helping me out with my questions on ACT-R software.

This dissertation would have remained unfinished without the support of my wife Dr. Olivia Das. I take this opportunity to express my gratitude to Olivia for her support. My seven year old Annika Mahamaya Das-Pal should also be credited in getting me complete this work—everyday, Annika would remind me that although she has finished her Kumon assignments, I am yet to finish my dissertation!

Finally, I thank York University, CUPE 3093 and NSERC for funding.

Table of Contents

List of Tables

List of Figures

Chapter 1
Introduction

The *goal* of this thesis is to develop simulation and closed-form cognitive models of learning interactive layouts.

How much practice is needed by a learner to progress from novice to expert level for an interactive layout? How effortful is it to learn a layout relative to other layouts? Answers to these questions are important in the design and evaluation of user interfaces. In this thesis, I look for cost-effective solutions to these questions.

1.1 Learning in User Interface Design

Learning refers to the acquisition of skill over time. Learning provides performance improvements with practice (Ritter et al., 2013). Yet, individuals often *forget* important skills due to disuse over time. This leads to decrease in performance. For example, an alarming 75% of 120 occupational first responders had their proficiency degraded only 6 months after receiving cardiopulmonary resuscitation training (McKenna & Glendon, 1985). Training and education are designed to improve learning and produce qualified performance through retention of knowledge (Kim, Ritter & Koubek, 2013).

Kim, Ritter and Koubek (2013) divide *learning* into three stages—Stage I (early stage), Stage II (intermediate stage), and Stage III (late stage). Stage I refers to the first stage where the user *acquires* declarative knowledge to perform a task—that is, enough knowledge to generate behaviour using the declarative knowledge structure, like following a recipe (Kim & Ritter, in press). Stage II refers to the second stage for *consolidating* the acquired declarative knowledge to procedural knowledge. Stage III refers to the final stage for *tuning* the existing declarative and procedural knowledge—users still get faster at the task, although the improvements get diminishingly smaller (Ritter et al., 2013). Kim, Ritter and Koubek (2013) illustrated how the shape of the *learning curve* looks like, reflecting all the three stages (Figure 1, p. 24). Figure 1.1 replicates this figure here. The thick continuous line in the figure indicates performance improvement through continuous practice over time.

Ritter et al. (2013) emphasize that the study of learning is important for the design of interactive layouts. Learning curve prediction can provide answers to several important questions related to the design of layouts—for example, it can provide insight into how fast item acquisition can be at a given stage of learning, which stage a learner is in, and how much practice is needed by a learner to reach the expert level. These answers may save valuable training time and cost and help to allocate resources effectively.

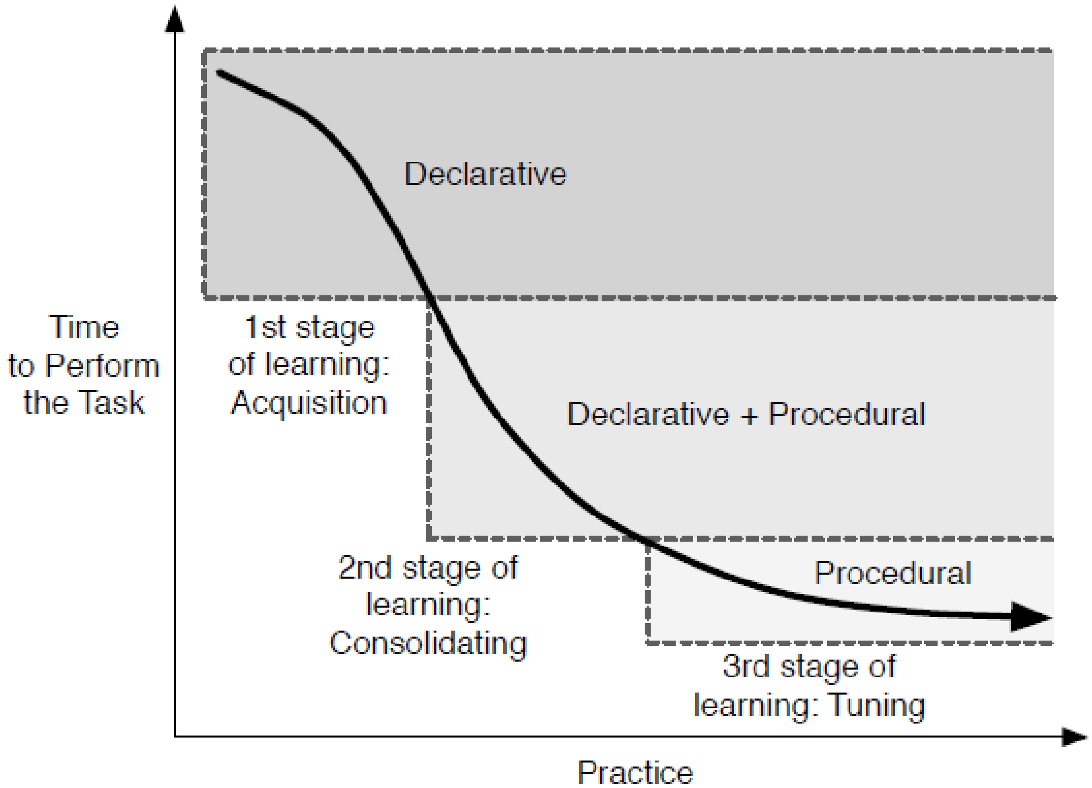

Figure 1.1 Performance change in three stages with declarative, mixed (i.e. declarative + procedural), and procedural representation of knowledge. The thick continuous line indicates continuous practice. (Figure taken from Kim, Ritter & Koubek, 2013).

My *first model* attempts to predict such a learning curve for text entry on a traditional phone keypad. It proposes a mechanism to account for the effect of the users' *visual exploration strategy* on task completion time when a learner progresses from novice to expert level. Although I demonstrate the use of this mechanism in text copying on a phone keypad, the mechanism may also be used for item acquisition on other kinds of interactive layouts.

Ritter et al. (2013, p. 137) suggests that for an interface, a relatively shallow learning curve but with a low intercept indicates that the interface may be *easier* to relearn after forgetting (see Figure 1.2a). Such a learning curve may be appropriate for interfaces that are not used often. In contrast, a relatively steep and long learning curve but with a lower final time may be *harder* to learn. Such a learning curve may be appropriate for interfaces that are used by experts (see Figure 1.2b).

If we were to compare a set of learning curves, one alternative is to use a generic curve fitting equation. However, a generic curve fitting equation is not derived from any principles of cognition (Busemeyer & Diederich, 2010; Chapter 1). Consequently, the effect of a cognitive phenomenon cannot be interpreted from such an equation. On the other hand, my *second model* being based on the ACT-R cognitive theory of declarative memory can be helpful in this case.

My *second model* attempts to *quantitatively* compare a set of learning curves of interfaces in terms of the *mental effort* needed to learn them.

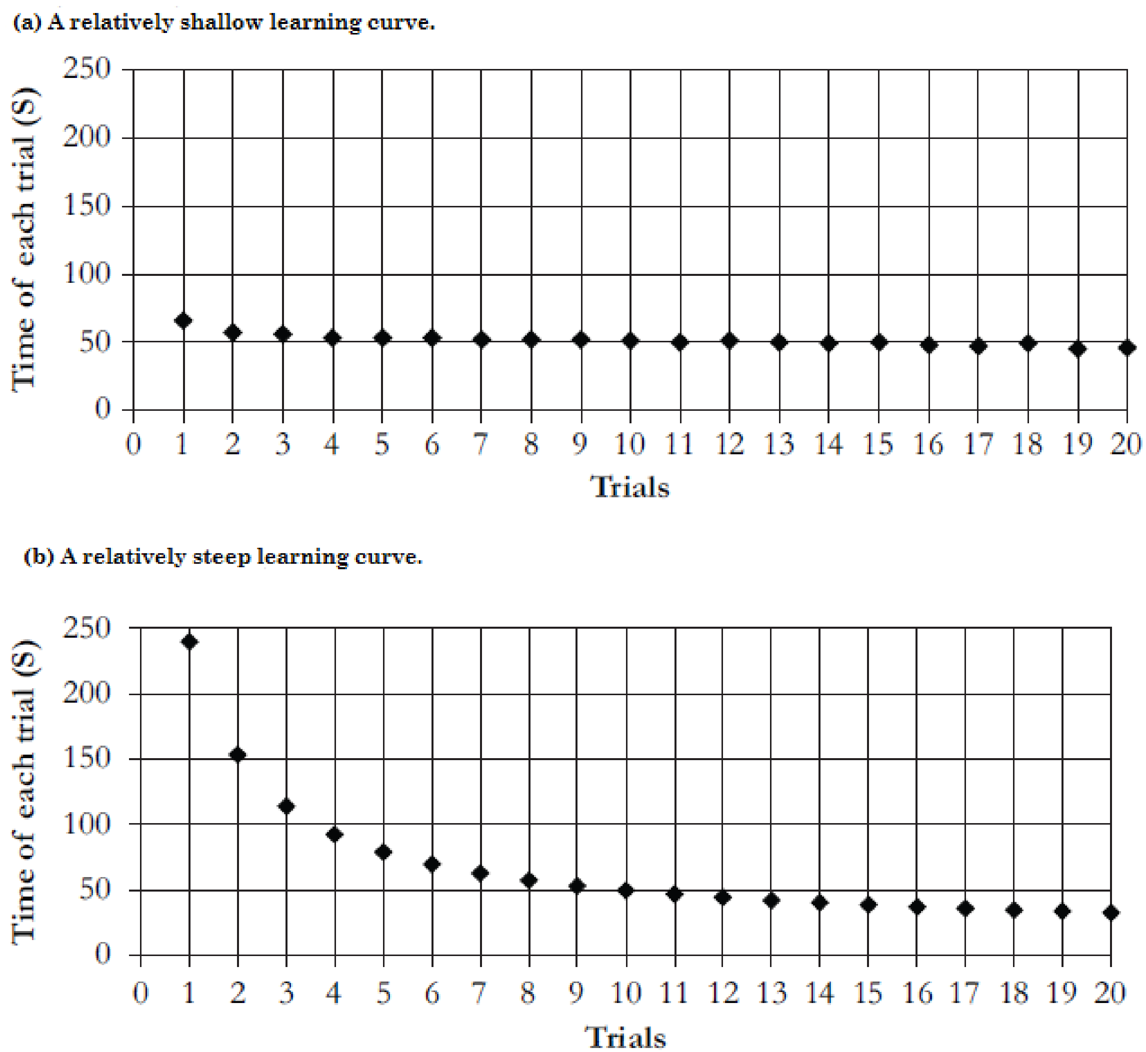

Figure 1.2 **Two learning curves with approximately the same final time: (a) a shallow learning curve versus (b) a steep and long learning curve. A relatively shallow learning curve but with a low intercept indicates that the interface may be *easier to relearn* after forgetting. Such a learning curve may be more appropriate for interfaces *not used often*. Contrariwise, a relatively steep and long learning curve but with a lower final time may be more appropriate for interfaces used by *experts*—interfaces that may be harder to learn and relearn but may be faster under continuous practice. The figure is taken from Ritter et al. (2013, p. 137).**

1.2 Model based evaluation in User Interface Design

Model based evaluation has potential advantages over experimental evaluation (Ivory & Hearst, 2001). Some of the advantages include (i) a reduced cost of evaluation; (ii) cost-effective comparison among alternative designs; and (iii) a reduced need of evaluation expertise through automation of some aspects of evaluation. Pew and Mavor (2007, as cited in Paik, Kim, Ritter, Morgan, Haynes & Cohen, 2010) encourage using models as shared representation that may help identify, predict, and when possible, mitigate risks. They point out that models have proven to be useful in predicting and preventing human or monetary losses.

Simulation modelling involves emulating the behaviour of a system over time. It involves designing and implementing a model of the system and executing the model on a computer. It is routinely applied while designing integrated circuits to predict device performance (Freed & Remington, 2000). However, it is infrequently used while designing human-machine systems. An important reason is that using available simulation modelling frameworks to predict human performance requires a great deal of expertise, time and labour to prepare the formal descriptions of the procedures (i.e. *how-to* knowledge) for operating in the domain of interest (Freed & Remington, 2000).

To make modelling usable by user interface evaluators with a range of expertise, it is necessary to provide easy-to-understand high-level abstractions. Such abstractions can replace cryptic, low-level descriptions of simulation models in parts or in its entirety, but can still predict the human performance. *Such high-level abstractions may expose fewer details about the underlying processes involved in generating behaviour but may be less complex and more straightforward to apply in comparison to low-level descriptions of simulation models.*

Overall, the necessity of simple and transparent high-level abstractions in cognitive modelling (e.g. Paik, Kim, Ritter et al., 2010) is the *primary* reason motivating my research. I present this thesis in two parts.

In the first part of the thesis, I develop a cognitive simulation model that executes a text copying task on a traditional phone keypad. The model provides insight into the amount of practice needed by a learner to progress from novice to expert level. I use a *mathematical equation* as a sub-model to emulate changes in the visual exploration strategy, as a learner progresses from novice to expert level. I do so to avoid developing a relatively complex, low-level description of visual search that would otherwise be required by current cognitive architectures such as ACT-R[1]. I augment this mathematical equation to a simulative sub-model that simulates the memory encoding process and the memory retrieval process. The resulting hybrid model predicts the time to find a symbol located on a button of the keypad, as the learner transitions from novice to expert level.

In the second part of the thesis, I develop a *closed-form* model that can assist in comparing the *mental effort* required to learn different layouts. These layouts vary

[1] As an example, the reader may refer to the complex low-level description of the custom visual search functionality in Ehret (1999). This custom functionality was built using an early version of ACT-R notation.

in terms of their *label representativeness*. Each label representativeness condition imposes a certain level of *difficulty* during learning. The model is computationally inexpensive, less complex and more straightforward to apply than an analogous simulation model.

1.3 Summary

This dissertation presents simulative and closed-form cognitive models of learning interactive layouts. I develop two models as part of this research.

First Model

My first model is a simulation model that emulates novice to expert transition in layout learning. Specifically, it simulates a text copying task on a traditional phone keypad. It models the change in visual exploration strategy from *search* of items to *choice* of items through a mathematical equation.

My first model can be generalized to any user interaction that involves progression of a learner from a phase that mostly involve *visual search* to a phase that mostly involve a *choice* decision. This type of interaction is common in user interfaces, for example selecting an item on the desktop, or a smart-phone, or a button-panel.

Second Model

My second model is a closed-form model that accounts for the combined effect of *practice*, *decay*, *proactive interference* and *mental effort* on task completion time. The main advantage of my model is that it can be used to *quantitatively* compare the *mental effort* required to learn different interactive layouts that vary in their *label representativeness*.

I have validated my model against two different sets of empirical data of graphical layouts. Although I have demonstrated the use of my model for graphical layouts, it can be generalized to compare effortfulness of different kinds of interactive layouts.

1.4 Organization of the dissertation

This dissertation is organized as follows. Chapter 2 provides a literature review. I briefly touch upon the previous works on *expert* performance, *novice* performance, *novice-to-expert* transition and, *effortful conditions* affecting learning. Chapter 3 presents a simulation model of novice to expert transition in text entry. The goal of chapter 3 is to predict how the time *to visually explore* a layout for a symbol affects the total time spent in copying it. Chapter 4 presents a closed-form model that accounts for *proactive interference* as well as *mental effort* in a combined fashion. The goal of chapter 4 is to develop a technique that can help to *quantitatively* compare the *mental effort* required to learn different layouts, which vary in terms of their *label representativeness*. Finally, chapter 5 concludes the dissertation with recommendation for future work. I also provide three appendices at the end. Appendix A provides a disclaimer. Appendix B explains the production rules that act as the procedural knowledge of my simulation model of Chapter 3. Appendix C is related to the closed-form model of Chapter 4. It shows sample computations of the predicted task completion time using the closed-form model.

Chapter 2
Literature Review

I develop two cognitive models. With respect to my *first* model, I focus on the *learning* of interfaces taking into account the transition from the behaviour of *search* at the novice level to the behaviour of *choice* at the expert level by a learner. With respect to my *second* model, I focus on comparing different *effortful conditions* of learning interfaces. There already exists relevant literature that analyzes the *learning* of interfaces. Also, there is previous work that analyzes the effect of *effortful conditions* in which learning takes place. These two sets of literature provide the context in which I develop my two models. I review previous works related to *expert* performance, *novice* performance, *novice-to-expert* transition, and some sample causes of *effortful conditions*. I also discuss a single example of work that observed the effect of effortful conditions on *retention* and *relearning*. I discuss a hypothesis called *Soft Constraint Hypothesis* that conjectures how performance cost can be interpreted in terms of *effort*.

Later in this dissertation, I develop a closed-form model for comparing effortful conditions of learning. I account for the effect of proactive interference in that model. For this, I briefly review the phenomenon of proactive interference in the domain of spatial learning. Thereafter, I discuss few earlier works on modelling the effect of interference. Next, I briefly review ACT-R theory—the theory of cognition that both of my models are based upon. Finally I briefly review Fitts' Law, which I use to predict the average movement time for a finger or a mouse cursor.

2.1 Expert Performance

The KLM performance model of Card et al. (1983, as cited in Cockburn & Gutwin, 2010) is the earliest response time model in Human-Computer Interaction (HCI). The KLM performance model predicts performance times for low-level human activities. It does so by decomposing tasks into component parts and applying standard predictions for each part. It predicts expert performance of routine tasks (e.g. prediction of the time to highlight and delete the word *color*) by summing the times for the task's atomic components: $T_{execute} = T_k + T_p + T_h + T_d + T_m + T_r$, where k refers to pressing a key or button, p refers to pointing with a mouse to a target object on a display, h refers to homing[2] on the keyboard or any other device, d refers to drawing a line segment on a grid, m refers to mentally preparing to do an action or a closely related series of primitive actions, and r refers to the system response during which the user has to wait for the electronic system she is interacting with. Default estimates of average values are used for some of the atomic components. For example, T_h = 0.4 sec is used for moving hand from keyboard to mouse or vice versa; T_m = 1.35 sec is used as the mental preparation time (Card et al., 1983). The values for T_k and T_p are often custom computed. T_k is computed in terms of seconds per keystroke. The expert-level value is chosen for T_k and T_p after substantial practice of the task in question (Kim & Ritter, in press). The KLM model thus predicts a single point of performance time (Cockburn & Gutwin, 2010). It does not model the transition from novice to expert behaviour (Cockburn & Gutwin, 2010, p. 13:6).

[2] A *home* location implies a location where a user will recoil and rest her finger after pressing a key on a keyboard/keypad. *Homing* refers to the process of recoiling to said home location.

Other work has also modelled the response time of expert performance in layouts. Some of these approaches include text entry in mobile computing (MacKenzie & Soukoreff, 2002), cell phone menu interaction (St. Amant, Horton & Ritter, 2007), and interaction with HTML mock-ups of interfaces where expert performance is estimated using the KLM model (John, Prevas, Salvucci & Koedinger, 2004). In the domain of mobile text entry using traditional phone keypad, Dunlop and Crossan (1999), Silfverberg et al. (2000), James and Reischel (2001), Butts and Cockburn (2002) and Cockburn and Siresena (2003) analyzed expert users. All of them reported single-point expert performance time.

2.2 Novice Performance

Pavlovych and Stuerzlinger (2004) presented an empirical study for learning of a phone keypad for text entry. The authors performed multiple studies on the behaviour of novices in text entry on a traditional phone keypad. I focus here on their first empirical study that involved a text copying task. This study measured the mean response time in *finding* a *letter* on a key of the keypad. This response time did not include the movement time of a finger from key to key (a motor component). From the empirical data, Pavlovych and Stuerzlinger inferred that a typical novice user will initially use a *search* strategy to locate the next *letter* to be entered, and later, after having *learned* the location of the letter, may use a *recall* strategy. Pavlovych and Stuerzlinger extrapolated a later part of their novice empirical data (that consisted of response times for a few sessions) based on a power function and thereby predicted a learning curve (p. 357). Their power function was however not based on any theory of human cognition.

Before Pavlovych and Stuerzlinger (2004), there were couple of empirical approaches that dealt with novice performance in text entry on the traditional phone keypad. To the best of my knowledge, they include the work of Dunlop and Crossan (2000, as cited in Cockburn and Siresena, 2003), James and Reischel (2001), Butts and Cockburn (2002), Cockburn and Siresena (2003), and Pavlovych and Stuerzlinger (2003). All of these works focussed on single-point novice performance time.

2.3 Novice-to-Expert Transition

Cockburn et al. (2007a)

Cockburn et al. (2007a) proposed a closed-form model of learning of a traditional graphical menu, i.e. a non-hierarchical, column layout of graphical buttons. The model incorporated a time component for visually *searching* a button by a novice user—this component is a function of the *number of buttons* on the layout; a time component for *choosing* a button by an expert user—this component is also a function of the *number of buttons* on the layout; and finally a *expertise* component that modelled the gradual change from novice to expert behaviour—assuming a spatially stable layout, this component is a function of the *number of trials previously completed* to select a button on the layout. The model is thus a function of the *number of buttons* and the *number of trials*—it does not take human cognition into account (Cockburn & Gutwin, 2010; p. 13:5).

Kim and Ritter (in press)

Kim and Ritter (in press) observed novice to expert transition in a spreadsheet task to examine learning. The task consisted of subtasks such as opening a file, performing calculations, typing name, inserting rows, inserting date using a command, and finally saving the work in a printable format. They examined two independent groups of subjects—one group completed the spreadsheet task using a mouse and a keyboard (*mouse group*) and the other group completed the spreadsheet task using *only* the keyboard (*keyboard group*). Both the groups completed a series of learning sessions for four consecutive days from Day 1 to Day 4. Kim and Ritter observed that *practice leads to faster task performance* for both the groups.

2.4 Effortful conditions

The causes for *effortful* learning conditions can be manifold. In this section, I review a few of them.

2.4.1 Effortful conditions due to difference in label representativeness

Ehret (2002) and Cockburn et al. (2007b)

Ehret (2002) observed that the response time in learning varies depending on the *representativeness* of the labels on the objects in a layout. In his experiment, he varied the representativeness of the labels on 12 stable graphical buttons that were arranged along the periphery of a circle on a computer screen. There were multiple label conditions differing in the *level* of representativeness. Listed in order of decreasing label representativeness, three of the conditions were: a *textual label* condition where the buttons were labelled with different colour names in English; an *arbitrary label* condition where the buttons were labelled with different, meaningless arbitrary icons that have nothing to do with colours; and an *invisible* *label* condition where the buttons were left blank with no labels at all. The task was to find a button with a pre-cued colour, from among the twelve buttons. Tooltips were available for the buttons, each tooltip revealing the colour for the button. Subjects were encouraged to use the tooltip, if memory failed.

Cockburn, Kristensson, Alexander and Zhai (2007b) performed an empirical study similar to that of Ehret (2002). They varied the label representativeness of the keys on a virtual keyboard. The representativeness was varied between a *labelled* (visible) condition and an *unlabelled* (invisible) one.

Both Ehret (2002) and Cockburn et al. (2007b) observed the following: the *higher* the label representativeness of buttons on a layout, the *lower* is the mental effort required to learn it. In contrast, the *lower* the label representativeness of buttons on a layout, the *higher* is the mental effort required to learn it.

The endeavour of Ehret (2002) and Cockburn et al. (2007b) motivates me to develop a cost-effective mechanism that could help to *quantitatively* compare the effortful conditions due to the difference in label representativeness of interfaces.

2.4.2 Effortful conditions due to difference in input modality

<u>Keyboard versus Mouse</u>

Through a spreadsheet task, Kim and Ritter (in press) observed how the differences in effortfulness of input modalities could affect learning. The task consisted of subtasks such as opening a file, performing calculations, typing name, inserting rows, inserting date using a command and finally saving the work in a printable format. They examined *input modality* as an independent factor on learning—the keystroke-driven modality (*keyboard group*) requiring keystroke commands and the mouse-based menu-driven modality (*mouse group*) requiring menu-driven commands. Keystroke-driven modality represented a *higher effort condition* compared to the menu-driven modality.

Subjects completed a series of practice sessions for four consecutive days from Day 1 to Day 4. During every practice session, the subjects had access to a study booklet to learn the task knowledge. Each practice session was allowed a maximum of 30 minutes. For example, on Day 1, subjects had a maximum of 30 minutes to study and perform the task.

Kim and Ritter observed that the keyboard group (*high effort condition*) was *slower* to complete the task on Day 1 than the mouse group (*low effort condition*). However it gradually became *faster* ending with a lower final time on Day 4 in comparison to the mouse group. This implies that keystroke-driven modality encouraged memory-intensive strategy, which resulted in a faster task time in a later stage of learning. This is in contrast to the mouse-based menu-driven modality that encouraged interaction-intensive strategy.

Command-line interface versus Direct manipulation interface

O'Hara and Payne (1998) suggest that a high effort condition demands *higher degree of planning* compared to the low effort condition. A higher degree of planning promotes better problem solving strategies than a lower degree of planning. They observed this in solving the 8-puzzle task using a command-line interface versus a direct manipulation interface. The command-line interface represented a *higher effort condition* compared to the direct manipulation interface. The 8-puzzle task involves sliding eight numbered tiles in a 3×3 matrix to reach a given solution state. One group of subjects practiced on the command-line interface (*command-line group*) and another group of subjects practiced on the direct manipulation interface (*direct manipulation group*). They observed that *lower number of moves* was taken to reach the solution state by the command-line group as opposed to the direct manipulation group. They concluded that the use of command-line interface led to a *higher degree of planning* as opposed to the use of the direct manipulation interface. This implies that the command-line interface (*high effort condition*) encouraged memory-intensive strategy, which resulted in *greater savings* in the number of moves taken to reach the solution state. This is in contrast to the direct manipulation interface (*low effort condition*) that encouraged interaction-intensive strategy.

2.4.3 Effortful conditions due to difference in system delay

A study by Golightly, Hone and Ritter (1999) used the 8-puzzle task to compare a *speech interface* and a *direction manipulation interface*. The speech interface involved a system delay, which had a disruptive effect on the interaction. This system delay resulted in making the interaction through the speech interface *more effortful* than the direction manipulation interface, which did not involve any system delay.

There were 10 different starting configurations for the 8-puzzle. All subjects received the starting configurations in the same order. One group of subjects used the speech interface and the other used the direction manipulation interface. There was one practice session. The session ended when a subject either completed all the 10 8-puzzles or had worked for 55 minutes, whichever came first.

Direct manipulation users were required to click on the tile they wished to move. If the tile was next to a space (i.e. the move was legal) the tile would move into the space. Speech interaction users, instead of clicking on the tile, indicated the tile they wished to move by vocally stating the digit labelled on the tile. There were no reliable differences between the interfaces in terms of the total task completion time.

Speech interaction users showed *longer move intervals* than direct manipulation users. This indicates that a *higher degree of planning* was undertaken by speech interface users *to accommodate the system delay* in comparison to the direct manipulation interface users. Moreover, speech interface users required a *lower number of moves* to reach a solution.

They concluded that the use of the speech interface required a *higher degree of planning* in contrast to the direct manipulation interface. This implies that the speech interface (*high effort condition*) encouraged a memory-intensive strategy, which resulted in *lowering the number of moves* taken to reach the solution state. This is in contrast to the direct manipulation interface (*low effort condition*) that encouraged an interaction-intensive strategy.

2.5 Effortfulness, Retention and Relearning

Through a spreadsheet task, Kim and Ritter (in press) observed how differences in effortfulness of input modalities can affect *retention* and *relearning*. They examined two independent factors on learning. The *first factor* was *input modality*—the keystroke-driven modality and the mouse-based menu-driven modality. Keystroke-driven modality represented a *higher effort condition* compared to the menu-driven modality. The *second factor* was *retention interval*—the retention interval being a period of inactivity between the last day of practice and the return day for the retention test. There were three retention intervals—a 6-day retention interval, a 12-day and an 18-day one.

There were 6 groups, each consisting of 10 subjects, randomly assigned. The 6 groups were divided into three pairs—Group1 and Group2; Group3 and Group4; Group5 and Group6. In each pair, one group used keystroke-driven modality (keyboard group—Group1, Group3, Group5) and the other group used the mouse-based menu-driven modality (mouse group—Group2, Group4, Group6).

Subjects completed a series of practice sessions for four consecutive days from Day 1 to Day 4. During every practice session, the subjects had access to a study booklet to learn the relevant task knowledge. After Day 4, Group1 and Group2 returned for the retention test on Day 10 (6-day retention interval), Group3 and Group4 returned for the test on Day 16 (12-day retention interval) and Group5 and Group6 returned for the test on Day 22 (18-day retention interval). In each test session, the subjects completed the spreadsheet task *without* the aid of the study booklet.

Kim and Ritter observed that the keyboard group was slower to complete the task on Day 1 than the mouse group. However it gradually became faster, ending with a lower final time on Day 4 in comparison to the mouse group.

Under the 6-day retention interval, the mouse group was observed to forget more than the keyboard group. That is, the mouse group showed more increase in time to complete the task. Under the 12-day retention interval, the keyboard group was observed to forget more than the mouse group. Under the 18-day retention interval, the keyboard group was again observed to forget more.

After the retention test on Day 10, Group1 and Group2 returned again for a relearning test on Day 16. On Day 16, the mouse group, Group2, showed greater decrease in task completion time compared to the keyboard group, Group1. Consequently, Kim and Ritter concluded that Day 10 may have served as a relearning opportunity and Group 2, a mouse group, relearned quickly.

From their study, Kim and Ritter suggest that the *high effort* condition of the keystroke-driven modality promotes a memory-intensive strategy which in turn facilitates *short-term retention*. On the other hand, the *low effort* condition of the mouse-based menu-driven modality promotes an interaction-intensive strategy. Such a strategy facilitates *long-term retention*. Moreover, a *low effort* condition promotes *quick relearning*.

2.6 Soft Constraint Hypothesis

Norman (1988, as cited in Kim and Ritter, in press) introduced the terms knowledge-in-the-world and knowledge-in-the-head to illustrate a fundamental design principle for an interactive system. Norman (1988, as interpreted by Kim and Ritter, in press) suggests that placing the knowledge in the environment (i.e. *in the world*) might be helpful in reminding, rather than placing the knowledge in memory (i.e. *in the head*). Anderson (1991, as cited in Fu & Gray, 2004) proposed a theory of *rational analysis*, which conjectures that goal-directed actions are chosen and executed through interactions between the human's adaptive mechanisms and the environment in ways that optimize efficiency.

As an extension to the above theoretical accounts, Gray and associates (Fu & Gray, 2001, 2004; Gray, Sims, Fu, & Schoelles, 2006) coined the term *soft constraints hypothesis* to provide an understanding of how cognitive resources are allocated in interactive behaviour. The *soft constraints hypothesis* conjectures that a mixture of four *effort* components may be needed to acquire knowledge in-the-world or to retrieve the knowledge in-the-head. The four effort components are *perceptual-motor search effort*, *perceptual-motor access effort*, *memory encoding effort*, and *memory retrieval effort* (Fu & Gray, 2004; p. 366).

The four *effort* components are described as follows:

When an item is at an unknown location, effort is needed to do *perceptual-motor search* to locate the item. This effort is the overall *effort* expended in activities such as planning, search strategy, spatial judgement, evaluation of items, and the actions carried out during the search for the target information.

When an item is at a known location, effort is expended for *perceptual-motor access* to reach the item. Examples of *access* include an eye movement to an icon in a menu ribbon of Microsoft Word or moving the mouse and clicking on a key of an on-screen keyboard.

To *store* an item in-the-head, an effort is required to encode it into the memory—*memory encoding effort*. To *use* an item present in-the-head, an effort is required to retrieve it from the memory—*memory retrieval effort*.

The mixture of the aforementioned four *effort* components mentioned above is allocated for interactive behaviour in a way that the least-effort path of executing the spatial task at hand, gets implicitly chosen (Fu & Gray, 2001, 2004). As acquisition of information from the environment (*the-world*) becomes *difficult*, people get motivated to choose the least-effort option of retrieving the information from memory (*the-head*), even if the retrieval is imperfect. Conversely, when acquisition of information from the environment becomes *easier*, people get motivated to choose the least-effort option of accessing information from the environment.

In their analysis, Fu and Gray (2001) accounted for one combination of the effort components—*the perceptual-motor access effort* + *the related memory encoding effort* + *the related memory retrieval effort*. However, they ignored the other combination—

the perceptual-motor search effort + *the related memory encoding effort* + *the related memory retrieval effort* (Fu & Gray, 2001, p. 112; Fu & Gray, 2004, p. 366). The reason for doing this is that there was a training phase (i.e. *non-expert* phase) before the actual empirical study (Fu & Gray 2004; p. 366). Hence they assumed that the second combination—*the perceptual-motor search effort* + *the related memory encoding effort* + *the related memory retrieval effort*—has already been met before their actual empirical study began.

Fu and Gray (2001, 2004) thus accounted for the combination of effort components that are expended predominantly in the *expert* phase of the learning curve. Unlike Fu and Gray, I focus on the combination of effort components that are expended predominantly in the *non-expert* phase of the learning curve.

2.7 Interference Phenomenon

Forgetting occurs not only due to passage of time but also through interference from information learned at other times (Wickens & Hollands, 2000, p. 252). Proactive interference (PI) is one explanation for the phenomenon in which encoding of non-target items prior to the encoding of target item disrupts the subsequent retrieval of the target item (Underwood, 1957; Keppel & Underwood, 1962).

Proactive Interference effect on spatial learning

Elmes (1988)

Elmes (1988) demonstrated that PI effects are relevant for spatial memory tasks. He used a variation of the card game known as *concentration* for the purpose. The subjects saw a tableau of cards face down. The game then involved turning over pairs of cards, one pair at a time. If the cards in a pair match (e.g. a king and a king), the cards remained exposed in their tableau positions until the end of the trial. If the cards in a pair did not match (e.g. a king and a queen), an error was recorded and the non-matching cards were turned back over with their faces down in their same tableau positions. A trial ended when the entire deck was matched. Then the cards were turned face down again, and the process was repeated. Learning was complete when the subject could expose the cards in matching pairs without error.

Elmes divided his subjects into 3 groups—2 experimental and 1 control. He made the control subjects learn just *one* game. In contrast, he made one group of the experimental subjects learn *two* successive games and made the other group of experimental subjects learn *four* successive games. Thus there were zero proactive games for the control subjects, and one or three proactive games for the experimental subjects before the terminal game. The same deck was used for all the games. Before each game the deck was thoroughly shuffled, which resulted in an essentially random placement of pair locations in each game. *For each group, once the terminal game was learned, there was a retention interval of 10 minutes, and then the terminal game was played again.* The experimental subjects committed more errors during the replay of the terminal game than the control subjects. Also, the experimental subjects with three proactive games committed more errors than the experimental subjects with one proactive game during the replay of the terminal game. From the result, Elmes concluded that *the lower the number of proactive tasks, the lower is the build up of PI on the target task* (*i.e. the terminal task*).

Modelling Interference

Despite advocating PI as the cause of forgetting, Keppel and Underwood (1962, Experiment 3) had suggested the *spontaneous recovery* of previously *extinguished* distractors as a cause while explaining the effect of the retention interval (see Altmann & Schunn, 2002 for details). Looking at the retention interval as a *passage of time*, Altmann and Schunn (2002) rationalized the spontaneous recovery of previously extinguished distractors *as* the loss of memory activation of the target item with passage of time, in other words the decay of the target item. Based on this, Altmann and Schunn (2002) developed a mathematical model that took into account the effect of both decay and proactive interference on verbal learning.

Other works attempt to model interference as a whole, not specifically proactive interference. I list some of those endeavours below.

West, Pyke, Rutledge-Taylor and Lang (2010) modelled the effect of interference on verbal learning using ACT-R. They modelled the role of interference on the *fan effect* using a single model parameter from ACT-R, the latency exponent. The *fan effect* refers to the phenomenon that cues associated with more facts result in slower recall of the target fact compared to cues associated with less. Although the value of the latency exponent parameter in ACT-R is traditionally expected to stay fixed across experimental conditions being compared, West and colleagues had modelled the interference effect in two different conditions (*false cues* versus *true cues*) with two different values of the latency exponent. Having identified this as an exception, they recommended investigation of an explicit model of the interference phenomenon in

ACT-R (p. 280). They pointed out that such a model should ideally keep the latency exponent parameter fixed across different experimental conditions being compared.

Spacing effect refers to the effect—of spacing practice events over a time span—on learning and retention. In their models of the spacing effect, Raaijmakers (2003) as well as Pavlik & Anderson (2005) accounted for interference in verbal learning. Both however abstracted the effect of interference using a constant, since their main focus was the spacing effect and not the interference phenomenon.

The models discussed above are all in the domain of *verbal learning*. While some of them have accounted for interference, only one of them has accounted for *both* interference and decay (Altmann & Schunn, 2002). On the other hand, there has not been much progress towards theoretically accounting for *interference* on *spatial learning*. The proactive interference due to distractors on spatial learning can be substantial, as Elmes (1988) had observed. In this thesis, I attempt to model the combined effect of *proactive interference* and *decay* in *spatial learning*.

2.8 ACT-R Theory

I develop models based upon a cognitive architecture known as ACT-R (Anderson et al., 2004). ACT-R is a unified theory of cognition, in the spirit proposed by Newell (1990), in that it reflects declarative and procedural learning and declarative forgetting (Kim, Ritter & Koubek, 2013; p. 23). ACT-R is designed to predict human behaviour by processing information and generating behaviour (Ritter & Kim, 2010).

The ACT-R system is composed of memory, perceptual, and motor modules. The memory module consists of a procedural memory sub-module and a declarative memory sub-module. The procedural memory sub-module consists of a set of production rules (procedures with an if-then structure) and a computational engine for interpreting those rules. The production rules coordinate cognition, perception and motor actions. The declarative memory module stores *chunks*. Each *chunk* represents the memory encoding of an object, and has an *activation* (i.e. a strength) associated with it. A chunk can be created, retrieved or updated by the production rules. The activities of the memory modules together with the actions of the perceptual and motor modules enable ACT-R to simulate cognition.

I develop two models—one simulative and one closed-form. In my first model, which is simulative, I use the declarative learning and forgetting mechanisms of ACT-R. Here, I leverage the default forgetting mechanism implemented through a constant representing the decay of memory. I utilize the procedural module for its computational engine to interpret the custom production rules of my simulation model.

For my second model, I focus solely on the set of ACT-R equations that describe the declarative memory strength as a function of practice. My use of ACT-R declarative memory equations as a stand-alone unit, while *abstracting out* the production rules, is not an exception. It follows previous work of Pavlik and colleagues (e.g. Pavlik & Anderson, 2005; Pavlik, Presson & Koedinger, 2007) on modelling spacing effects and Altmann and Schunn (2002) on modelling proactive interference.

The core of ACT-R declarative memory builds upon the notion of memory activation. This notion posits that *chunks* (memory encodings of objects) have different levels of activation to reflect their past use: chunks that have been used *recently* or chunks that are used *frequently* receive a high activation. This activation decays over time if the *chunk* is not used. When the cognitive system needs to retrieve a *chunk*, memory returns the one with the highest activation at that instant. The job of memory retrieval is complicated by the noise in activation levels, which can temporarily make a *chunk* more active than the current one, or which can temporarily push all *chunks* below a threshold, thereby making the cognitive system transiently unable to recall information. Furthermore, the activation of a *chunk* controls its speed of retrieval. These dynamics bear similarity to other formal activation constructs (e.g., Just & Carpenter, 1992; as cited in Anderson et al., 2004).

ACT-R theory consists of independent sets of equations, each set driving the computation for the relevant ACT-R module. In the following subsections, I discuss the three core equations behind the ACT-R declarative memory module.

2.8.1 ACT-R Activation Equation of Declarative Memory

The equation describing the *activation* of a *chunk* in the memory is given by

$$A_{n+1} = B_n + O_{n+1} \qquad \textbf{Activation Equation}$$

In the above equation, A_{n+1} is the activation of the *chunk* during its $(n+1)^{th}$ practice. B_n is the base-level activation of the *chunk* after n practices have been completed—B_n is computed just before the $(n+1)^{th}$ practice happens. O_{n+1} denotes the optional terms in the equation. The optional terms are accounted for when a practice is in progress. Thus, O_{n+1} is accounted for when the $(n+1)^{th}$ practice is in progress. One such optional term is the *noise* component. Noise is assumed to cause transient fluctuations in activation levels. In this dissertation, I account for noise *only* in the first model (the simulation model). I do not account for noise in my second model (the closed-form model). To keep both of my models simple, I do not account for any other optional terms. Note that avoiding the use of optional terms is not an exception. It follows previous work of Altmann and Schunn (2002) on modelling proactive interference and Cochran, Lee and Chown (2006) for modelling the arousal effect.

2.8.2 ACT-R Base-Level Activation Equation of Declarative Memory

The equation describing the base-level activation of a *chunk* in the memory is given by

$$B_n = ln\left(\sum_{j=1}^{n} t_j^{-d}\right) \qquad \textbf{Base-Level Activation Equation}$$

where n is the number of practices of the *chunk* completed so far, t_j is the age of the j-th practice of the *chunk*, and the negative exponent $-d$ is a constant that controls how quickly activation decays. B_n is computed just before the $(n+1)^{th}$ practice. As postulated by ACT-R theory, the *negative d* term models the loss of memory strength with the passage of time. The equation therefore represents the strength of a *chunk* as the sum of a number of individual memory strengthenings, each corresponding to a past practice event. It implies that each time a *chunk* is practiced, the activation of the *chunk* receives an increment in strength that decays away as a power function of time.

Overall, B_n is the strength of a *chunk* (memory encoding of an object) after n practices of the *chunk* have been completed. A practice of a *chunk* is said to occur whenever the *chunk* is presented to the declarative memory. Such presentation happens due to either visual encoding or recall of the *object* represented by the *chunk*.

The base-level activation equation is a central theme of my research. It postulates the metaphor that information is lost from human memory due to decay, a process indexed by time. Several other researchers (for example, Peterson & Peterson, 1959), have also postulated the hypothesis of memory weakening due to decay. In contrast, another school of researchers have historically argued that interference from distracting information is an important cause of forgetting (for example, Keppel & Underwood, 1962).

2.8.3 ACT-R Reaction Time Equation of Declarative Memory

The time required for the declarative memory to respond to a request for a *chunk* representing an object is given by the following equation:

$$RT_{n+1} = I + Fe^{(-f*A_{n+1})} \qquad \textbf{Reaction Time Equation}$$

In the above *reaction time equation*, RT_{n+1} is the reaction time of the $(n+1)^{th}$ practice. RT_{n+1} depends on the activation A_{n+1} of the *chunk* being practiced. I is an intercept time reflecting the fixed time cost of visual encoding and motor response (Anderson et al., 2004, p. 1043). F is the latency factor, and maps the activation to time. f is the latency exponent. The *reaction time* does not depend on the estimation of the parameters I and F. The effect of I and F is only to scale the critical quantity $e^{(-f*A_{n+1})}$ onto the range of the latencies.

The fixed time cost of a visual encoding is set at 85 ms which is taken from the estimate used by ACT-R for human attention to move to an object at a given location (Anderson et al., 2004; p. 1039).

The time cost of a motor response is set according to the task specific behaviour. Different values are chosen depending on whether the movement is, for example, a finger press on a key of a computer keyboard, or pointing with a mouse and then clicking a button in a graphical user interface.

2.9 Fitts' Law

Fitts' law predicts the Movement Time MT it takes for a pointing device (e.g. a finger or a mouse cursor) to move a given distance to an item of a given size. It is expressed as follows.

$$MT = a + b * \log_2\left(\frac{A}{W} + 1\right) = a + b * ID \qquad \textbf{Fitts' Law (MacKenzie's formulation)}$$

In the above equation of Fitts' law (MacKenzie, 1992), A is the amplitude of the movement (e.g. the distance between two keys on a keyboard—a source key where the movement begins from and a target key where the movement ends), and W is the width of the target item. The log term in the equation is called the *index of difficulty* ID. ID is measured in bits.

In target acquisition tasks on user interfaces, the width W of the target item is measured assuming that the item is either rectangle or square shaped (MacKenzie and Buxton, 1992) or assuming a bounding rectangle around the item if it is not rectangular (Silfverberg et al., 2000, p. 12). According to MacKenzie and Buxton (1992, p. 221), the *smaller* of the two sides of the rectangle seems more indicative of the accuracy demands of a target acquisition task in a user interface. Hence MacKenzie and Buxton recommends to consider the smaller side of the rectangle to be the target width W.

There are other versions of Fitts' law such as the original Fitts' formulation or Welford's formulation whose ID term is different from that of MacKenzie's formulation. Whenever A/W ratio drops below 0.5, these formulations result in a negative ID. However, to predict the movement times on human-computer interfaces including phone keypads and graphical user interfaces, MacKenzie and Buxton (1992, p. 219) and Kim and Ritter (in press) recommend the use of MacKenzie's (1992) formulation because it prevents the ID from being negative.

The coefficients a and b are usually determined empirically for a given device (e.g. computer screen, phone keypad, computer keyboard) and the interaction style (e.g. pointing with a mouse cursor, pointing with a finger, pressing with a thumb) (Pavlovych & Stuerzlinger, 2004; p. 352). They are determined by regressing observed movement times on the *index of difficulty* ID (Mackenzie, 1992; p. 98).

For a given interface and interaction style, the Fitts law coefficient *a* and *b* are held constant as noted above. In this situation, change in movement time MT depends only on the change in ID.

In the early and intermediate stages of spatial learning, the movement time is only a small fraction of the total time needed to perform a target acquisition. The rest of the time is spent in non-movement tasks such as search, encoding, and recall (e.g. Salthouse, 1986; Pavlovych & Stuerzlinger, 2004; Kim, Ritter & Koubek, 2013; Kim & Ritter, in press). If we consider a given mean ID in a location learning task on a stable user interface and same interaction style, the movement time as predicted by Fitts' law stays the same over practice sessions. However the non-movement time of item acquisition should decrease with decrease in search time and improvement in recall over practice sessions.

In this thesis, I consider the *novice-to-expert* transition phase. For the simplicity of my analysis, I include an *average movement time* predicted from Fitts' law in the task completion time (to acquire items on a layout) as and when required. To do so, I take into account the minimum and the maximum amplitude possible on the layout. Next, I show a sample calculation of movement times using Fitts' law.

Given a *task completion time* (to acquire an item on a layout), I often refer the movement time (e.g. to move a finger or a mouse cursor) predicted from Fitts' law as *Fitts time*. I refer to the non-movement time portion that remains after subtracting the Fitts time from the task completion time as *non-Fitts time*. As noted earlier, the *non-Fitts time* is assumed to be spent in activities such as search, encoding, and recall.

Prediction of an average movement time using Fitts' law: An example

I now show how Fitts' law can be used to predict *average movement time* for text input on a traditional phone keypad using either left or right thumb. Figure 2.1 shows a traditional phone keypad of Nokia 5190 cell phone. I use the Fitts' law coefficients a = 0.176 sec and b = 0.064 sec/bit. Silfverberg et al. (2000) determined the values of these coefficients empirically for one-handed thumb use for text entry. Silfverberg et al. had used the traditional keypad of a Nokia 5100 series cell phone for collecting human data.

Figure 2.1 Traditional keypad layout as found on a Nokia 5190 cell phone. Letters occupy eight keys. They are spread over key-2 to key-9.

I first digitize the screenshot of the keypad provided in Silfverberg et al. (2000). Here, I use the Engauge Digitizer version 4.1 for digitization. Using the digitizer, I set the height of a key as 1 unit. Then, in terms of the height of a key as 1 unit, I obtain the approximate distances between the centers of three pairs of keys—the vertical distance between key-1 and key-4, the horizontal distance between key-1 and key-2, and the diagonal distance between key-1 and key-9. The minimum and the maximum among these three distances are the minimum and the maximum amplitudes respectively. In this case, the minimum amplitude is 1.44 units (vertical distance between key-1 and key-4) and the maximum is 5.30 units (diagonal distance between key-1 and key-9). Next, using Fitts' law in the MacKenzie's formulation equation, I obtain a minimum $MT = a + b * \log_2(A/W + 1) = 0.176 + 0.064 * \log_2(1.44/1 + 1) \approx 0.258$ sec and a maximum $MT = a + b * \log_2(A/W + 1) = 0.176 + 0.064 * \log_2(5.30/1 + 1) \approx 0.346$ sec. Then the average movement time is approximated as $\text{MT} = (\text{minimum MT} + \text{maximum MT}) / 2 \approx 0.302$ sec.

2.10 Summary

The goal of this thesis is to develop simulation and closed-form cognitive models for learning of layouts. Through a brief review of literature, this chapter creates a context to develop these models. In this regard, I discussed related work for *expert* performance, *novice* performance, and *novice-to-expert* transition.

Then I discussed some sample *effortful conditions*. I also discussed the effect of effortful conditions on retention and relearning. Moreover, I discussed the *soft constraint hypothesis* that conjectures how performance cost can be interpreted in terms of *effort*.

Next, I briefly reviewed the phenomenon of interference, specifically the phenomenon of *proactive interference* in the domain of spatial learning. Thereafter, I discussed earlier work on modelling the effect of interference.

I briefly reviewed the ACT-R theory that my models are based upon. The ACT-R theory provides a simulation framework of mutually interacting modules of cognition. This enables the creation of simulation models that can explain aspects of novice to expert transition in layout learning. The ACT-R theory provides a rich set of mathematical equations that models declarative memory. These equations can help to create closed-form models accounting for the combined effect of *practice*, *decay*, *interference* and *mental effort*.

Finally I reviewed Fitts' Law. I use it to predict the average movement time for a finger or a mouse cursor.

Chapter 3
A Simulation Model of Novice to Expert Transition in Layout Learning

3.1 Introduction

The work presented in this chapter is related to the peer-reviewed material of Das and Stuerzlinger (2007, 2008).

In layout learning, the involvement of cognitive and perceptual processes is substantial, especially in the early and intermediate stages of learning (e.g. Kim, Ritter & Koubek, 2013). This is evident in text copying tasks using keyboard layouts (e.g. Salthouse, 1986; John, 1996) or in the item acquisition tasks on graphical layouts (e.g. Byrne, 2001; Ehret, 2002; Kim & Ritter, in press). The time for item acquisition on a layout can be divided into two parts—non-movement time and movement time. Ahlstrom et al. (2010, p. 1374) suggests that, unlike experts, who spend most of the time on movement aspects, non-experts spend the majority of the time in the *visual search* for items.

I will use the term *non-Fitts time* to refer to the non-movement time. The *non-Fitts time* (NFT) is the part of the user's task completion time that remains after subtracting the movement time.

Goal

The goal of this chapter is to predict the learning curve for a text copying task on a traditional phone keypad. This would need the prediction of *non-Fitts times* for the novice to expert transition. Specifically, the aim is to account for the effect of the *visual exploration* behaviour on the *non-Fitts time*.

I represent the *non-Fitts time* (NFT) to be the sum of two parts—one part is the *Visual Exploration Time* (VET) and the other part is the *Non-Exploration Time* (NET), that is, NFT = NET + VET. I describe VET and NET next.

I conjecture that VET is either the *visual search time* for a symbol present on a button (i.e. symbol location) of a layout or, the *choice time* for a button on the layout or, a combination of both. Here, the *visual search time* is the time to search for a symbol by a pure novice. The maximum VET is the *visual search time* and the minimum VET is the *choice time*. VET is thus a continuum from the *visual search time* to the *choice time*. I obtain the VET from a mathematical equation.

I conjecture that NET is either the *visual encoding time* of a symbol (and its location) or the memory *retrieval time* of a symbol (and its location).

To meet the goal, I develop a model that simulates the task of copying textual symbols by pressing buttons (labelled with the symbols) on a traditional phone keypad layout. My model is able to simulate the different stages of learning. Figure 3.1 illustrates the model. My model has two sub-models: a *simulative sub-model* based on the ACT-R 6.0 simulation framework to predict NET and a *non-simulative* *sub-model* to predict VET. The results of the *simulative sub-model* are utilized in the calculation of VET.

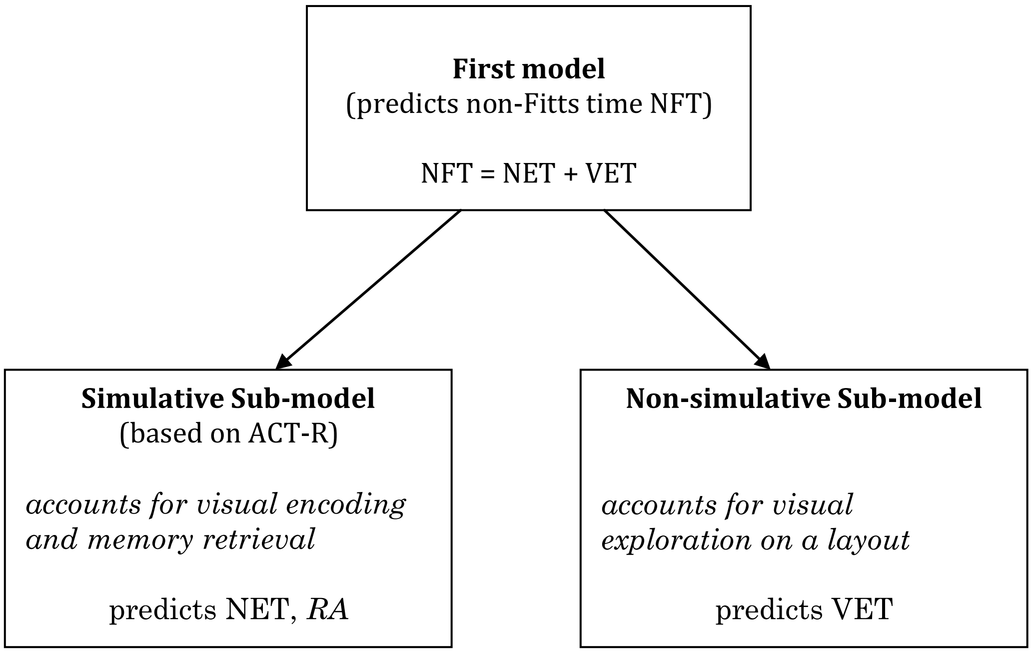

Figure 3.1 My first model

Motivation

I am motivated to develop the simulation model due to the following reasons. First, Cognitive simulation can help to predict the learner's future cognitive states (Kim, Ritter & Koubek, 2013; p. 23). A cognitive simulation model can be developed within a cognitive architecture such as ACT-R. A model thus developed is believed to simulate the interactions of cognitive subsystems and predict human performance accurately (Kim, Ritter and Koubek, 2013; p. 23).

Second, the modelling endeavour can reduce the cost of running experiments with subjects. Once a model is validated with experimental data, the validated model can provide predictions of human performance, reducing the cost to evaluate systems and interfaces (see for example, Pew and Mavor 2007, St. Amant et al. 2007 as cited in Kim, Ritter and Koubek, 2013).

The third reason that motivates me to develop the model is specific to a constraint in the classic ACT-R Theory. The classic version of ACT-R theory realizes a vision subsystem that is a purely attentional system—that is, although the vision subsystem models the visual encoding time for a symbol as a fixed cost, it does not model any visual search strategy or any mechanism to assess visual search cost. Previously few works (e.g. Byrne, 2001; Ehret, 2002) have tried to alleviate this constraint of classic ACT-R by implementing a custom visual search functionality. However such an endeavour may need a great deal of expertise in specifying cryptic, low-level descriptions of simulation models within a cognitive architecture. Consequently, I model the *visual exploration time* VET in terms of a mathematical equation and avoid implementing a custom simulation model for visual search. Although a custom search model (such as the ones by Byrne, 2001; Ehret, 2002) may provide a richer description of visual search strategies, my mathematical equation is less complex and more straightforward to apply.

3.2 Text Entry on Cell Phone

The simulation model I develop in this chapter is for copying textual symbols on a traditional cell phone keypad. My model predicts the novice to expert transition. To do this, one of the things my model needs is the human non-Fitts times for the first few sessions.

In this section, I mention some empirical studies that have been carried out in the domain of text entry on cell phones. These studies observed text entry performance for novice or expert or both.

To the best of my knowledge, Dunlop and Crossan (1999) were the first to investigate text entry on cell phones. Shortly after that, Silfverberg et al. (2000) performed an empirical study and provided a model to predict text entry speed of expert users. This was followed by other studies: one by James and Reischel (2001) and another by Butts and Cockburn (2002).

Dunlop and Crossan (1999) and Silfverberg et al. (2000) concentrated on expert users. Dunlop and Crossan (2000, as interpreted in Cockburn and Siresena, 2003) as well as Pavlovych and Stuerzlinger (2004) concentrated on novice users. A few studies such as the ones by James and Reischel (2001), Butts and Cockburn (2002), as well as Cockburn and Siresena (2003) analyzed both novice and expert users.

The studies by James et al. (2001) and Butts et al. (2002) point out that the model of Silfverberg et al. (2000) is an overly optimistic model, as it focuses solely on the motor part. Silfverberg et al. (2000) effectively ignores any potential cognitive component, which is non-zero even for expert behaviour (James and Reischel, 2001). Pavlovych and Stuerzlinger (2004) then empirically demonstrated the existence of this cognitive component in novice user behaviour through a text copying task on a traditional phone keypad.

Related to the work of Pavlovych and Stuerzlinger (2004), I have obtained the human non-Fitts times for the first few sessions through personal communication with Dr. Andriy Pavlovych. I will test the novice predictions of my model against this human data.

3.3 The task to be executed by the model

A single run of my model executes the *task* of copying a group of 5 distinct English letters in a given session, for 160 sessions. At *every session*, a group of 5 distinct English letters out of 26 letters are randomly chosen and copied by the model. The copying task is performed on a simulated layout of the traditional phone keypad labelled with English letters as shown in Figure 3.2. For the ease of explanation, I assume that there are only three main areas on the visual scene that I use for my model. They are, *text display area*, *text output area* and *keypad area* from top to bottom respectively. Figure 3.2 shows the three areas.

At a given session, the five letters to be copied are first displayed in the *text display area*. To accomplish the task, the model looks at the letter to be copied in the *text display area*; next it shifts its attention to the target letter on the *keypad area*; finally the model presses the key containing the letter. As a consequence of pressing the key, the letter is outputted in the *text output area*.

A single model run predicts the mean non-Fitts time to copy a letter in each of the 160 sessions.

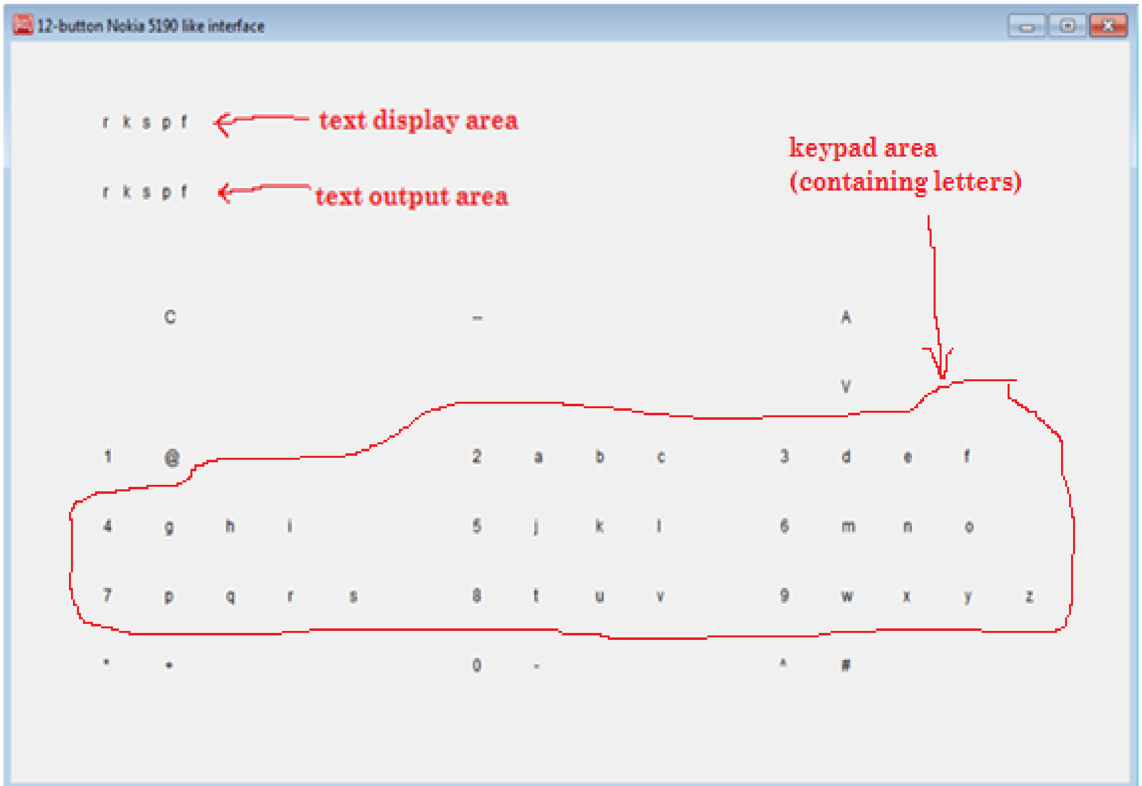

Figure 3.2 Layout showing *text display area*, *text output area* and *keypad area*. This layout acts as the visual scene for the simulation sub-model.

3.4 Model Foundation

The model that I introduce in this chapter predicts the non-Fitts time (NFT) to copy an English letter on a traditional phone keypad. It consists of two sub-models—one *simulative* and the other *non-simulative*. I explain the two sub-models next.

3.4.1 Simulative sub-model

The *simulative* sub-model predicts the followings. (i) The *Non-Exploration Time* (NET) to copy a letter in a session. (ii) The ratio of the number of successful memory retrievals in a session to the total number of retrieval attempts in that session. I call this ratio *recall accuracy* (*RA*). The *RA* will be used in the non-simulative sub-model.

To develop my simulative sub-model, I use the learning mechanism of ACT-R declarative memory. I use the ACT-R production rules as an engine to control the cognitive actions such as visual encoding of a symbol or retrieval of a symbol from the declarative memory. ACT-R theory assumes that a visual encoding of a symbol takes a constant time (Anderson et al., 2004). It also assumes that at any given time point along practice sessions, the retrieval time is the same for any symbol. Thus, at any given time point, the NET (the visual encoding time or the retrieval time) is the same for any symbol.

My simulation sub-model utilizes five modules of ACT-R 6.0. These modules are—the motor, vision, declarative memory, procedural memory, and goal modules. (i) I use the motor module to model the interaction of the right-hand thumb with the keys on a keypad. Figure 3.3 shows the model of the keypad that my simulation sub-model interacts with, through the motor module. It is a traditional keypad of the Nokia 5190 phone. (ii) I use the vision module to model the visual attention on the symbols. Figure 3.2, shown earlier, represents the visual scene in the external environment that my simulation sub-model interacts with, through the vision module. At the top of the visual scene is the *text display area*. The area displays the five letters to be copied. Below the text display area is the *text output area*. It shows the letters that has already been copied. The remaining area is the *keypad area*. The keypad area shows the characters on the traditional phone keypad of a Nokia 5190 phone. (iii) The declarative memory module stores information about symbols and their locations. It keeps track of activations of symbols. It models increase in activation due to practice and loss of activation due to decay. I model the noise in declarative memory by setting the *activation noise scale* parameter (*ans*) to a value of 0.1. I choose a small value for the noise to model a scenario where memory retrieval failures can normally occur as they do in real subjects. (iv) The procedural memory module controls the execution of my production rules. To keep my model simple, I create the production rules in a way that no two rules compete at any given instant of time. The rules execute a finite state machine. I do not add any noise to the procedural memory. An English description of the nineteen production rules that I create for my simulation sub-model is provided in Appendix B. (v) I use the goal module to keep track of the current state of the execution.

The key production rules of my simulation sub-model are as follows:

can-recall-letter-location-on-keypad matches if the keypad coordinates of the current letter (that has just been encoded from the *text display area*) is same as the information present in the retrieval buffer and fails to match if it doesn't. If the match occurs, the model will execute a motor action directly, without any attention shift, to enter the letter.

cannot-recall-letter-location-on-keypad matches if the keypad coordinates of the current letter (that has just been encoded from the *text display area*) is not same as the information present in the retrieval buffer (more specifically when the retrieval buffer is empty). If the match occurs, it will lead to the shift of visual attention, to the *keypad area*, for the current letter.

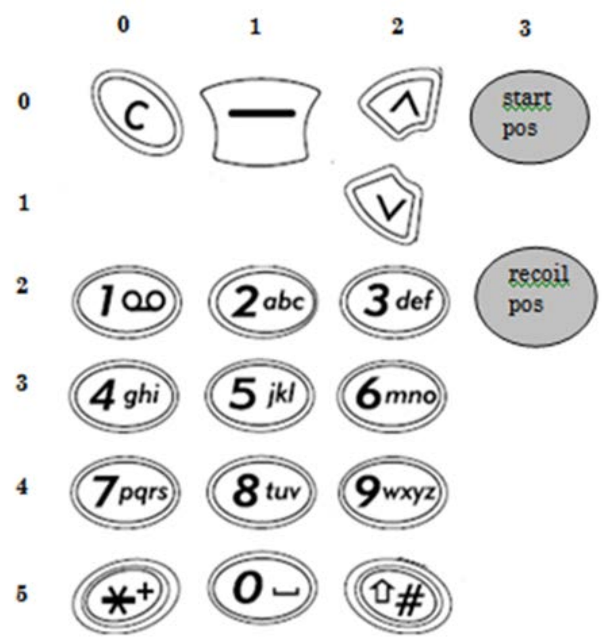

Figure 3.3 Virtual grid for the Nokia 5190 keypad.

Adapting the Motor Module of ACT-R 6.0 Framework

To support the development of my simulative sub-model for text copying on a traditional phone keypad, I add a model for the keypad interface of the Nokia 5190 phone to the motor module of ACT-R 6.0. As part of my model development, I add certain motor movement styles as follows:

(i) I create a virtual grid of key locations, the `start` position, and the `recoil` position for the right thumb. Figure 3.3 shows this grid, with four columns and six rows. Columns 0 to 2 contain the keys themselves, whereas column 3 contains the `start` position and the `recoil` position of the right-hand thumb. Although the recoil/home position might vary and hence affect the movement time predicted by Fitts' law, my assumption of a fixed recoil position is still valid for this work since I am interested only in the non-Fitts time portion of the user's task completion time. I further assume that a) all the keys on the keypad are of the same size, b) the width of a key is one *key unit*, c) the horizontal and vertical distance between adjacent keys on the keypad is one *key unit*, and d) that the user is right-handed (holds the phone handset in her right hand) and uses only the thumb to press keys.

(ii) I create a new movement style called thumb-recoil-to-location that models the movement of the right-hand thumb from a key to the recoil/home location (3, 2). The grammar of the ACT-R model description language for the new style is as follows:

```
+manual>
  ISA      thumb-recoil-to-location
  hand     right
  finger   thumb
  to-loc   location in virtual grid
```

(iii) The default *Peck* movement style of ACT-R (a directed movement of a finger to a new location followed by a keystroke, all as one continuous movement) may be considered sufficient for text entry modelling. However, this style was developed for computer keyboards where only one letter is mapped to one key. Since a single key on a traditional phone keypad contains multiple characters, I extend the ACT-R system to allow the modeller specify the location of the target key as well as the character the simulative sub-model would be pecking for. I name the new movement style `peck-to-location-for-char`. The grammar for the extended style is as follows:

```
+manual>
  ISA       peck-to-location-for-char
  hand      right
  finger    thumb
  to-loc    location in virtual grid
  for-char  string
```

(iv) The default *Punch* movement style of ACT-R (a down-stroke directly followed by an upstroke of a finger, for pressing a key that is already directly below the finger) was originally developed for the home keys (recoil / resting positions of the fingers) on a computer keyboard. In my case, however, punch can be executed on any key. I therefore extend the default movement style to enable the modeller to specify the character to be punched as well. I name the new style `punch-for-char`. The grammar for the extended style is as follows:

```
+manual>
  ISA       punch-for-char
  hand      right
  finger    thumb
  for-char  string
```

It is to be noted that among the three movement styles that I have described above, I have included the `punch-for-char` style for the sake of completeness of my simulative sub-model's description. The `punch-for-char` style does not actually get used in the execution of the task described in this chapter. This is because, as described earlier, the task executed by the simulative sub-model involves copying *distinct* English letters in a given session—there is *no repetition* of letters getting transcribed. Consequently, the production rule implementing the movement style `punch-for-char` does not get invoked during the task execution.

I leave the ACT-R motor module at its default configuration and computational logic, except that I force the Fitts' law mathematical function in the ACT-R motor module to return zero at every simulated key press. I do this so that the movement time predicted by Fitts' law does not get added up in the task completion time during simulation. This allows me to focus only on the non-Fitts time.

3.4.2 Non-simulative sub-model

The non-simulative sub-model predicts the *Visual Exploration Time* (VET) to copy a letter in a session. To develop this sub-model, I make the following conjecture—when the user is a pure novice with respect to a given keypad layout, she performs an explicit *visual search* to find a letter. However, as she gains expertise with practice over time, she starts spending less time in visual search; she now begins to spend more time *choosing* letter location (i.e. button). The non-simulative sub-model accounts for this gradual transition from a *searching* process to a *choosing* process.

I represent the non-simulative sub-model as a mathematical equation. I predict VET from that equation. VET is either the *visual search time* for a letter present on a button of a keypad or, the *choice time* for a button or, a combination of both *visual search time* and *choice time*. The novice VET is the *visual search time* and the expert VET is the *choice time*. VET is thus a continuum from the *visual search time* to the *choice time*.

In the computation of VET, I account for the *choice* behaviour of an expert user via Hick's law (Hick, 1952). Guided by Sears et al. (2001, p. 161), I treat Hick's law as a non-cognitive model that predicts the *choice time* for a button as a function of the number of known alternative buttons.

In the computation of *visual exploration time* VET, I further use the *recall accuracy* term noted earlier. *Recall accuracy* (*RA*) is the ratio of the number of successful memory retrievals in a session to the total number of retrieval attempts in that session. *RA* is predicted by the simulative sub-model. *RA* influences the gradual shift of behaviour from pure *search* for letters at novice level to pure *choice* of buttons at expert level as the user learns the keypad layout with practice. *RA* is one term that accounts for the effect of cognition in the non-simulative sub-model.

3.4.2.1 Novice Visual Exploration Time

I assume that the *novice* VET to find a letter on a layout is the time required to find it in the *first session*. I denote the novice VET by the *visual search time* VST.

The empirical data that I validate my model against does not specify the VST. Instead it provides the *non-Fitts time* NFT of a digit and the NFT of a letter in the *first session*. Specifically, these NFTs came from unpublished data. I obtained this data through personal communication with Dr. Andriy Pavlovych related to the work of Pavlovych and Stuerzlinger (2004). I estimate the VST of a letter from these two NFTs as described next.

The empirical *non-Fitts time* NFT of a letter in a *session* was obtained as an average of the total non-Fitts time spent in copying 5 consecutive letters from an external reference in that session. Similarly, the empirical NFT of a digit in a *session* was obtained as an average of the total non-Fitts time spent in copying 5 consecutive digits from an external reference in that session. The empirical NFT of a letter in the *first session* is 1748 ms and the empirical NFT of a digit in the *first session* is 974 ms.

I assume that the NFT of a *digit* was observed to be *smaller* than the NFT of a *letter* because of the following three reasons: First, participants were active users of the traditional phone keypad used for dialling phone numbers. Therefore, they were familiar with the digit locations on the keypad; Second, the digits on the keypad of a Nokia 5190 phone are substantially larger in size compared to the letters (see Figure 3.3); Third, unlike letters, only a single digit is mapped to each key (see Figure 3.3). Therefore, I assume that some form of "pop-out" effect (Treisman & Gelade, 1980) occurs for digits. These reasons permit me to speculate that the *visual search time* VST required for a digit is negligible compared to the VST required for a letter. As a consequence, I assume that the time required to copy a digit consists mostly of the NET component of the non-Fitts time.

To estimate the *visual exploration time* VET for a letter in the *first session* (i.e. VST), I assume the following. (a) I assume that at any given time point along the practice sessions, the time to visually encode a symbol into the ACT-R declarative memory *is the same* (specifically, 85 ms—an ACT-R axiom (Anderson et al., 2004, p. 1039)). (b) I further assume that a letter or a digit has been considered equiprobably in the context of the text copying task in Pavlovych and Stuerzlinger (2004). Therefore, in the *first session*, I assume that the time to retrieve a letter or a digit from the ACT-R declarative memory *is the same*. Thus, overall, in the *first session*, I assume that the *non-exploration time* NET (i.e. the time to visually encode or the time to retrieve a symbol from the ACT-R declarative memory) for either a digit or a letter *is the same*.

In summary, I assume the following. To *copy* a digit or a letter in the *first session*: (i) the NFT required to copy a digit is only the NET; the VST for a digit is assumed to be negligible compared to the VST for a letter, and is therefore ignored for my modelling purposes. (ii) the NFT required to copy a letter consists of the NET plus the VST. (iii) the NET is *same* for either a letter or a digit.

Thus for the *first session*, I can write,

$NFT_{digit} = NET_{digit}$ as per the assumption (i) above.

$NFT_{letter} = NET_{letter} + VST_{letter}$ as per the assumption (ii) above.

$NET_{letter} = NET_{digit}$ as per the assumption (iii) above.

Using (i), (ii) and (iii), I can derive,

$$VST_{letter} = NFT_{letter} - NET_{letter}$$

$$= NFT_{letter} - NET_{digit}$$

$$= NFT_{letter} - NFT_{digit} = 1748 - 974 = 774 \text{ ms}$$

Thus the *visual search time* (VST) for a letter (i.e. the VET for a letter in the *first session*) is approximately 774 ms.

3.4.2.2 Expert Visual Exploration Time

Guided by Cockburn et al. (2007a), I consider the *expert* VET to be the *choice reaction time* (CRT) for a button on a layout. I use Hick's Law (Hick, 1952) to compute the CRT. Hick's law is defined as follows.

$$\text{CRT} = a + b * \log_2 (n) = a + b * H \qquad \textbf{Hick's Law}$$

In the above equation, n is the number of already known buttons to choose from. The coefficients a and b are empirically determined constants. They are determined by regressing the observed choice reaction times on the *bits per stimulus presentation*, H (Seow, 2005, p. 324). The assumption is that users know the correct response (e.g. which button to press) for each stimulus (e.g. letter) (Sears et al., 2001, p. 160).

The coefficient b serves as an index of the time taken to process one bit of information (Seow, 2005, p. 320). The coefficient a reflects the individual differences in sensory-motor lags in task performance (Seow, 2005, p. 329).

Welford (1968, as cited in Soukoreff & MacKenzie, 1995) assumes that in continuous text-entry there is no uncertainty as to when the stimulus signal arrives. Consequently, Welford (1968, as cited in Soukoreff & MacKenzie, 1995) suggests to assume the coefficient a to be 0 for continuous text-entry.

Welford (1968, as cited in Soukoreff & MacKenzie, 1995) also maintains that the throughput (also known as *rate of gain of information* in Hick's paradigm (Seow, 2005, p. 332)) of key presses in response to stimulus presentation would range between 5 to 7 bits per second (see Sears et al., 2001, p. 160). I assume that the maximum choice processing throughput to be appropriate for a pure expert user. Therefore I set the constant b to 1/7 seconds per bit.

Sears et al. (2001, p. 161) suggested that, the number of alternatives (i.e. n) should be based upon the number of keys (i.e. reactions) on the keypad rather than the number of letters (i.e. stimuli). Hence I set $n = 8$ since the traditional phone keypad of Nokia 5190 phone has the *letters* spread only over *eight* buttons.

Consequently, my *choice reaction time* CRT for a button is

$$\text{CRT} = b \log_2 (n) = (1/7) * \log_2 8 \approx 429 \text{ ms.}$$

3.4.2.3 The Equation representing Non-Simulative Sub-Model

The equation representing the non-simulative sub-model is as follows:

$$\text{VET} = (1 - RA) * \text{VST} + RA * \text{CRT}$$ **Visual Exploration Time Equation**

In the visual exploration time equation, VET is the *visual exploration time,* VST is the *visual search time* (i.e. the *novice* VET), and CRT is the *choice reaction time* (i.e. the *expert* VET). The term RA represents *recall accuracy*.

Next I explain the way to compute RA specific to the task executed by my model.

As I had mentioned earlier, a *session* in a run of my simulative sub-model consists of the task of copying a group of 5 distinct English letters. In a given run, the task is repeated across 160 sessions; at *every session* a group of 5 distinct English letters are randomly identified out of 26 letters and copied by the model.

I have created my simulative sub-model so that during each run the sub-model first attempts to recall the location of a pre-cued letter on the keypad. The simulative sub-model does this through its system of production rules. Each recall attempt either fails or succeeds. A successful recall results in the retrieval of the chunk containing the location information of the letter. The retrieval of a chunk occurs when its *activation* exceeds the *retrieval activation threshold* set at the onset of the run. Let the number of successful recalls in a given session be x where $x <= 5$ (the total number of letters to be copied in a session being 5). Then I express *recall accuracy*, *RA* as follows:

$$RA = x / 5 \qquad \textbf{Recall Accuracy Equation}$$

The *recall accuracy RA*, thus, ideally varies from 0 corresponding to *visual search only* by a pure novice, to 1 corresponding to *choice only* by a pure expert.

The *visual exploration time equation* reflects the following: With practice, the user is able to know more and more letter locations on the keypad; hence her *visual search time* for a letter location decreases towards zero. With the increase in familiarity of keypad layout, she adapts her behaviour to spend more time in *choosing* a letter location (button) out of all the letter locations she knows so far, and consequently her *choice reaction time* dominates.

An equation similar to my *visual exploration time equation* was used earlier by Cockburn et al. (2007a, Equation 4). Cockburn et al. used it to model the increase in user's level of expertise during the novice to expert transition in learning a graphical menu. Assuming a spatially stable menu layout, their equation was a function of the *number of buttons* on the menu and the *number of trials previously completed* to select a button on the menu. Their equation, therefore, is *not* based on any cognitive principles (Cockburn & Gutwin, 2010, p. 13:5).

I model the user's level of expertise by the notion of *Recall Accuracy*. The *Recall Accuracy* computation is influenced by the activation equation of ACT-R's declarative memory. Since the said activation equation accounts for the effect of both learning (declarative) and forgetting (declarative), the *Recall Accuracy* therefore also reflects the effect of learning as well as forgetting unlike Cockburn et al. (2007a).

I substitute VST (=774 ms) and CRT (=429 ms) in the *visual exploration time equation* with the values obtained in earlier sections. Thereby I obtain an average visual exploration time (in ms) as follows:

$$\text{VET} = (1 - RA) * \text{VST} + RA * \text{CRT} = (1 - RA) * 774 + RA * 429$$

3.5 Non-Fitts Time Equation: the new Hybrid Model

My new model is a hybrid of the two sub-models—the simulative sub-model and the non-simulative sub-model described in the previous sections. At a high level of abstraction, my hybrid model can be symbolically represented as follows:

$$\text{NFT} = \text{NET} + \text{VET} \qquad \textbf{Non-Fitts Time Equation}$$

In the non-Fitts time equation above, NFT is the mean non-Fitts time, NET is the mean *non-exploration time* and VET is the mean *visual exploration time* per letter, corresponding to a given *session*. The VET equation expressed earlier consists of the terms VST, CRT and RA. The NFT equation can be thought of as a unification of the cognitive components (NET, *RA*) and the non-cognitive components (VST, CRT). NET and *RA* are predicted from the simulative sub-model.

I use the *Non-Fitts Time Equation* to model the user's non-Fitts time for copying a pre-cued letter in a given session. The *Non-Fitts Time Equation* can be rewritten as follows:

$$\mathrm{NFT} = \mathrm{NET} + \mathrm{VET}$$

$$\text{or,}\quad \mathrm{NFT} = \mathrm{NET} + (1 - RA) * \mathrm{VST} + RA * \mathrm{CRT}$$

$$\text{or,}\quad \mathrm{NFT} = \mathrm{NET} + (1 - RA) * 774 + RA * 429$$

where VET is substituted by an expression derived earlier.

3.6 Comparison of model data and human data

In this section, I validate the first 15 sessions of the predicted non-Fitts times against the first 15 sessions of the human non-Fitts times.

3.6.1 Human data to validate the model

The human data that I validate my model against came from the unpublished data that I obtained through personal communication with Dr. Andriy Pavlovych related to the work of Pavlovych and Stuerzlinger (2004). They measured the non-Fitts time to copy a visually pre-cued English letter on a traditional phone keypad of Nokia 5190 phone. The keypad was connected to a computer. The keypad is shown earlier in Figure 3.3. There were 12 participants in that study, recruited from university campus. Five participants were female, one was left-handed, and three were frequent users of text messaging. All had extensive computer experience (seven years or more). One did not own a cell phone. One reported using text messaging on the cell phone daily, another two did weekly; all others used it infrequently.

The data entry application used in Pavlovych and Stuerzlinger (2004) was created in a way so as to avoid repeated key presses required to arrive at a letter on a traditional phone keypad (Figure 3.3). For example, to copy the character sequence `cei`, the user needed to press the key containing `c` only once instead of pressing it thrice (refer to Figure 3.3 for the location of `c`), the key containing `e` only once instead of pressing it twice, and the key containing `i` only once instead of pressing it thrice.

To data-fit my model, I obtained the human mean non-Fitts times to copy a *letter* for the first 15 *sessions*. The non-Fitts time per *letter* in a *session* was obtained as an average of the total non-Fitts time spent in copying 5 consecutive letters in that session. Table 3.1 shows the human data points for the first 15 *sessions*. Figure 3.4 shows the plot. The standard deviations associated with the data points were not available to me.

Table 3.1 Human mean non-Fitts times to copy an English letter on traditional phone keypad of a Nokia 5190 cell phone.

Session	Mean non-Fitts time per English letter

1	1748
2	1618
3	1890
4	1811
5	1591
6	1608
7	1621
8	1691
9	1628
10	1651
11	1688
12	1617
13	1623
14	1798

15	1614

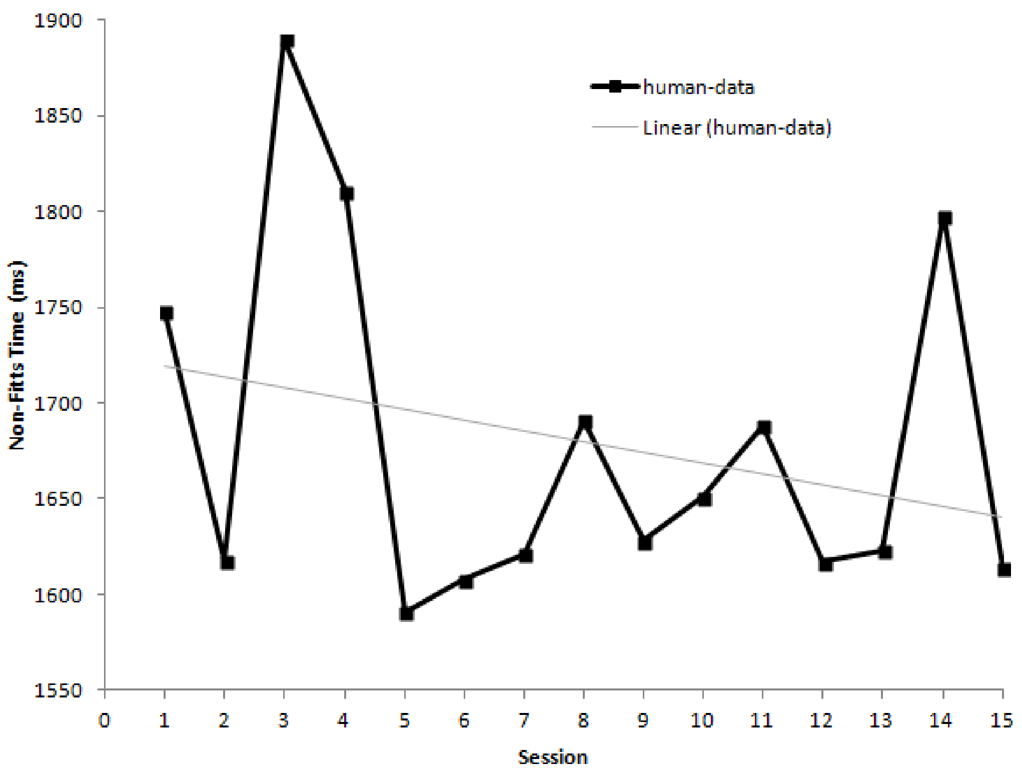

Figure 3.4 Human data for the first 15 sessions. The linear regression line for the human data is also shown.

I assume that the first 15 sessions of the human data in Figure 3.4 belong to the Stage I of learning (Kim, Ritter & Koubek, 2013). From Figure 3.4 it is evident that considerable oscillation exists in the human data from sessions 1 to 7 and sessions 13 to 15. This is possibly owing to this relatively short test in Stage I of learning.

Although correlation between the human data and model data is a way to show the degree of match between them (Grant, 1962, as cited in Ritter et al., 2011), in the present case a direct correlation between the human and the model data for the first 15 sessions makes little sense due to the oscillations in the human data. Given the situation, a comment from Taatgen and Van Rijn (2010, p. 251) may be relevant here:

"*When we create a cognitive model, it is not our goal to fit a particular data graph, although this may be part of the process, but to explain the phenomena that we are interested in.*"

An alternative in this case could be to try matching the rate of learning of the model data to that of the human data for the first 15 sessions. In this regard, I compute a linear regression line of the human data points using MS Excel. Figure 3.4 shows the linear regression line. The equation for the regression line is $\hat{Y}_h(X) = -5.66X + 1725$. Using this equation, I obtain the session 1 point as $\hat{Y}_h(1) = 1719$ ms and session 15 point as $\hat{Y}_h(15) = 1640$ ms. Thus the regression line of human data shows a difference of about $\hat{Y}_h(1) - \hat{Y}_h(15) = 1719 - 1640 = 79$ ms between session 1 and session 15.

<u>*Making the model output compatible with human data*</u>

To stay compatible with the scenario in which the human data was collected, I discard the model data for the very first group of 5 distinct letters at every model run. I assume that the *first session* for the model execution starts from the second group of 5 distinct letters to be copied. The reason behind this is explained below.

At every simulation run, the modelled recall accuracy is *always* 0 for the very first group of 5 distinct letters. This is because the model has been developed in such a way that at each run, the model is *not familiar* with the location of any letter that belongs to the first group of 5 distinct letters. However, I validate my model against a set of human data that has been obtained from a group of participants who were frequent or infrequent users of cell phone (Pavlovych & Stuerzlinger, 2004, p. 355). Hence I assume that a participant being familiar with the phone keypad layout, would have, on average, recalled the location of at least one letter while entering the first group of letters. I therefore assume that the mean human recall accuracy has *not* been zero for the very first group of 5 distinct letters with respect to a participant.

3.6.2 Power Analysis: number of simulation runs for 15 sessions

Before I try matching the model data to the human data, I need to estimate the *minimum number of runs* my model should execute to provide stable predictions of non-Fitts times for the first 15 sessions. I use Power Analysis (Howell, 2007) to obtain an initial estimate of this minimum number of runs.

In the power analysis, I need to consider an effect size of interest (Ritter et al., 2011, p. 114). The present case is a case of *matched samples* where session 1 and session 15 are being compared. The effect size *d* in case of *matched samples* is defined as (Howell, 2007; p. 223) follows:

$$d = \frac{expected\ difference\ in\ the\ means\ of\ two\ populations\ of\ observations}{standard\ deviation\ of\ the\ difference\ scores\ drawn\ from\ these\ populations}$$

The estimated effect size of interest $\hat{d}$ (Howell, 2007; pp. 189-190) can be computed as follows:

$$\hat{d} = \frac{\hat{Y}_{\mathrm{h}}(1) - \hat{Y}_{\mathrm{h}}(15)}{s_{1-15}}$$

where (i) the numerator represents the desired difference in mean non-Fitts time that my model should achieve between session 1 and session 15. This desired difference should be $\hat{Y}_{\mathrm{h}}(1) - \hat{Y}_{\mathrm{h}}(15)$ as obtained from the regression line of human data. (ii) the denominator s_{1-15} represents the sample standard deviation of difference scores. The difference scores can be obtained by subtracting the simulated non-Fitts time of session 15 from that of session 1 corresponding to a given run, for several runs. Agresti and Finley (1997, p. 180) specifies that for a sample size greater than or equal to 30, the sample standard deviation provides a good approximation for the population standard deviation. Therefore at this point, I decide to run my model 30 times to obtain s_{1-15}.

As noted earlier, the difference $\hat{Y}_{\mathrm{h}}(1) - \hat{Y}_{\mathrm{h}}(15)$ obtained from the regression line of human data is 1719 – 1640 = 79 ms.

I then ran my model for 30 runs and obtain a value of s_{1-15} = 142 ms. I ran the model on a Dell System XPS 15Z laptop running the 64 bit Windows 7 Home Premium operating system. 30 runs took about 3 minutes. The ACT-R parameters were *retrieval threshold* (rt) = 0.25, *latency factor* (lf) = 0.01, *activation noise scale* (ans) = 0.1 and *decay rate* (bll) = 0.5. The rest of the parameters were at their default values.

Thus the estimated effect size $\hat{d}$ is

$$\hat{d} = \frac{\hat{Y}_{\mathrm{h}}(1) - \hat{Y}_{\mathrm{h}}(15)}{s_{1-15}} = \frac{79\text{ ms}}{142\text{ ms}} = \mathbf{0.56}$$

For a matched-sample t-test, the non-centrality parameter δ (Howell, 2007; p. 224) will be as follows:

$$\delta = \hat{d} * \sqrt{N}$$

where N is the number of subjects. In present case, N would imply the minimum number of simulation runs required for a given value of δ and the desired effect size $\hat{d}$.

For a matched-sample t-test ($\alpha = 0.05$, two-tailed), the minimum δ should be 4.2 to achieve a power of 0.99 (Howell, 2007; p. 678). Therefore the minimum number of simulation runs required to achieve the desired effect size $\hat{d} = 0.56$ between sessions 1 and 15 is

$N = (\delta/\hat{d})^2 = (4.2/0.56)^2 = 57$

I should therefore run my model for a minimum of 57 times for the aforementioned desired effect size. However, I should repeat the runs until I see that the change in cumulative standard deviation (between run N and $N - 1$) and change in cumulative mean (between run N and $N - 1$) become negligible (Ritter et al., 2011).

I ran my simulation model 100 times for the first 15 sessions. Across those 15 sessions, I find that the absolute value of the change in the cumulative standard deviation between run 99 and run 98 is less than 4 ms (and therefore assumed negligible) and the absolute value of the change in the cumulative mean between run 99 and run 98 is also less than 4 ms (and therefore assumed negligible) in each session. Therefore, I conclude that to model the human data for the first 15 sessions, a minimum of 99 model runs is required.

I further carried out an analysis based on standard error of mean (SEM) recommended by Ritter et al. (2011) to find out the minimum number of runs that my model would need to obtain stable predictions of the mean and the standard deviations of non-Fitts times in every session across 160 sessions. I describe the SEM based analysis next.

3.6.3 SEM based analysis: number of simulation runs for 160 sessions

The central limit theorem states that given a population with mean μ and standard deviation σ, the sampling distribution of the mean (the distribution of sample means) will have a mean equal to μ and a standard deviation equal to $\sigma/\sqrt{N}$ where N is the size of each sample. The distribution will approach the normal distribution as N increases (Howell, 2007; p. 170). The standard deviation of this distribution of sample means is also known as the Standard Error of Mean (SEM).

When the population standard deviation σ is unknown, the sample standard deviation s is used as an estimate of σ for large sample size (Howell, 2007, p. 175). The Standard Error of Mean (SEM) then becomes

$$SEM = \frac{s}{\sqrt{N}}$$

The 95% confidence limits on the population mean is *sample mean* $\pm$ 1.96**SEM*. That is, the population mean has a 95% chance of being within the range of (*sample mean* $-$ 1.96**SEM*, *sample mean* $+$ 1.96**SEM*). Thus, one way to determine how many simulation runs are to be executed is to run the model until the estimated range of the population mean is small enough for my purposes (Ritter et al., 2011, p. 109).

For a spread of $\pm$25 ms of non-Fitts time with 95% confidence, we would have to have a SEM of 25/1.96 or a SEM of about 12.7 ms (25 = 1.96*SEM, or 25/1.96 = SEM ≈ 12.7).

In the present case, the NFT in each *session* in a given simulation run can be thought of as a *sample point*. Thus, each simulation run will generate one sample point per sample for 160 *matched samples*. Therefore N runs will generate N sample points per sample for 160 *matched samples*. Using the equation $SEM = \frac{s}{\sqrt{N}}$,, N can be found as $(s/SEM)^2$.

I find that the absolute value of the change in cumulative SD between run 99 and run 98 is less than 7 ms (and therefore assumed negligible) in each of the 160 sessions. Therefore for my modelling purposes, I decide to use the SD of the first 99 runs in each session to compute the number of runs N for that session. As an example, I show next how I compute the minimum number of runs required for session 1. In session 1, the SD of the first 99 runs is 103 ms. Therefore $\mathrm{N} = (s/SEM)^2 = (103/12.7)^2 \approx 66$ given that SEM ≈ 12.7 as computed earlier. Thus, a minimum of 66 runs is required to provide stable prediction in session 1. Table 3.2 shows the minimum number of runs required for every session for the first 15 sessions after SEM based analysis.

Table 3.2 Minimum number of model runs required for the first 15 sessions using SD of first 99 runs for each session. Obtained using SEM based analysis.

Session	Minimum number of model runs required
1	66
2	40
3	43
4	36
5	45
6	43

7	52
8	46
9	76
10	86
11	54
12	83
13	96
14	79
15	92

On inspecting the minimum number of model runs across all 160 sessions, I find that session 50 needs 322 runs. This is the maximum of the minimum number of runs computed across all 160 sessions. Since my model runs have been inexpensive, I decided to run my model **500 times**. Figure 3.5 shows the plot of modelled mean non-Fitts times from 500 runs across 160 sessions. The vertical SEM bars for each model data point are also shown in the plot. The shape of the plot is similar to the curve showing the three stages of learning in Kim, Ritter and Koubek (2013).

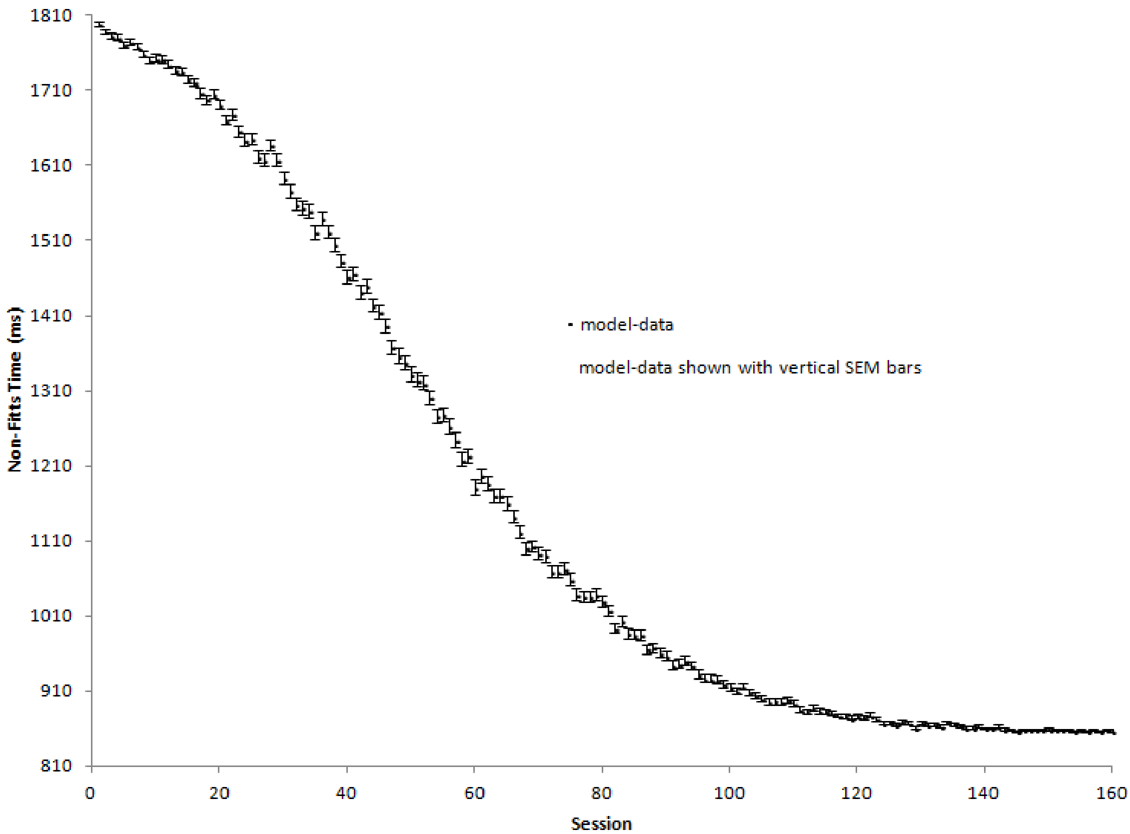

Figure 3.5 Modelled mean non-Fitts times over 500 runs across 160 sessions. SEM bars are shown on the model data points.

3.6.4 Model data versus human data for first 15 sessions

Now that I have obtained the minimum number of runs my model should execute to provide stable predictions of non-Fitts times, I go back to my earlier question—is the rate of learning of model data similar to that of the human data for the first 15 sessions? The answer to this question would help me verify whether my model follows the learning phenomena reflected by the human data of first 15 sessions. I test whether the slopes of the two linear regression lines—one from 15 human data points and the other from 15 model data points—are statistically significantly different or not. I describe this test next.

3.6.4.1 Testing the difference between the slopes of two regression lines

Table 3.3 tabulates the human data and the model data for the first 15 sessions. Figure 3.6 shows the plot. The model data for each of the 15 sessions is the average over 500 runs. The RMSE of the fit is 117 ms. The R^2 of the fit is 0.09. In Table 3.3, X denotes the session number, Y_h denotes the human non-Fitts time, and Y_m denotes the model non-Fitts time. In the descriptions that follow, the subscript h denotes *human* and the subscript m denotes *model*.

Table 3.3 Human data and model data for first 15 sessions. The model data for each session is the average over 500 runs.

X (session)	Y_h (human) (ms)	Y_m (model) (ms)
1	1748	1798
2	1618	1788
3	1890	1783
4	1811	1780
5	1591	1771
6	1608	1775
7	1621	1768
8	1691	1758
9	1628	1751
10	1651	1753
11	1688	1752
12	1617	1745
13	1623	1737
14	1798	1735

15	1614	1725

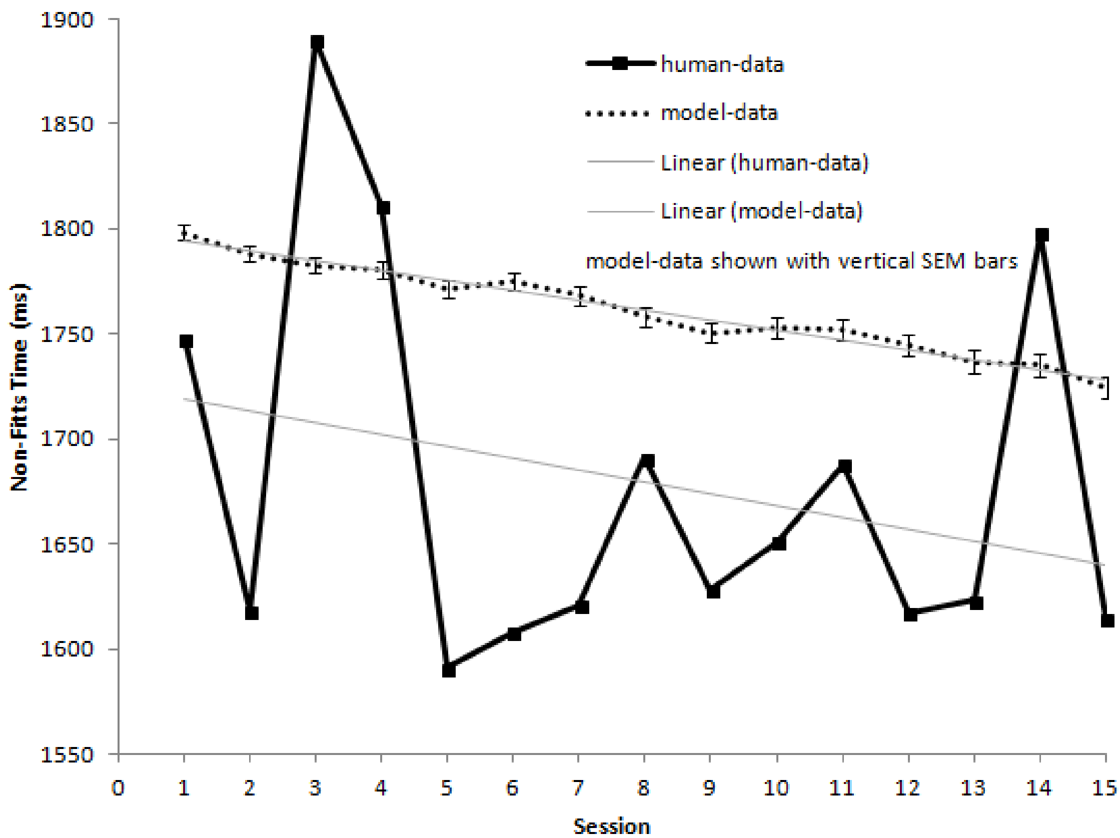

Figure 3.6 Human data and model data for first 15 sessions, as well as their linear regression lines. SEM bars are shown on the model data.

The equations of the linear regression lines for the human data and the model data are as follows.

Linear regression line (human)
$\hat{Y}_h = -b_h X + a_h = -5.66X + 1725$

Linear regression line (model)
$\hat{Y}_m = -b_m X + a_m = -4.73X + 1799$

Below I use the formula for the error variance taken from Howell (2007, pp. 244-245). The error variances for the human data and model data are as follows.

Error variance (human, N = 15)
$s^2_{Y_h \cdot X} = \frac{\sum(\mathrm{Y_h} - \hat{Y}_h)^2}{N-2} = 8198$

Error variance (model, N = 15)
$s^2_{Y_m \cdot X} = \frac{\sum(\mathrm{Y_m} - \hat{Y}_m)^2}{N-2} = 12$

The variance of X is as follows.

Variance of X (N = 15)
$s^2_X = \frac{\sum(\mathrm{X} - \bar{X})^2}{N-1} = 20$

The above tables can be summarized below as follows.

Human	Model
$b_h = -5.66$	$b_m = -4.73$
$s^2_{Y_h.X} = 8198$	$s^2_{Y_m.X} = 12$
$s^2_X = 20$	$s^2_X = 20$
$N = 15$	$N = 15$

The analysis in the rest of the subsection below follows Howell (2007, p. 258). It utilizes the mathematical formulae and follows the style of reporting results as recommended by Howell (2007, p. 258).

The *t* test for differences between two independent slopes is directly analogous to the test of the difference between the means of two independent samples (Howell, 2007, p. 258).

The Shapiro-Wilk test (a test for normality) revealed that the model data appears to be normally distributed, W(15) = 0.977, p = 0.946. But human data is not, W(15) = 0.824, p = 0.008 (Mayers, 2013, Chapter 3, pp. 50-51). Subsequently, I applied the z-score tests of skewness and kurtosis to the human data (Mayers, 2013, Chapter 3, pp. 52-54). The obtained z-score for skewness is 2.1 and the obtained z-score for kurtosis is 0.427. Since the z-score for skewness is close to 2 and the z-score for kurtosis is lower than 2, I conclude that the human data is reasonably normally distributed.

Howell (2007, p. 203) suggests that *for equal sample sizes*, violating the assumption of homogeneity of variances produces very small effects. In general, Howell notes that t-test is *robust* against the departures from its underlying assumptions.

Drawing from the conclusions above, I apply the t-test for differences between two independent slopes. In the ensuing discussions, the subscript h denotes *human* and the subscript m denotes *model*.

The null hypothesis that we test is H_0: $b_h \; of \; population = b_m \; of \; population$. If H_0 is true, the sampling distribution of $b_h - b_m$ is normal with a mean of zero and a standard error of

$$s_{b_h - b_m} = \sqrt{s_{b_h}^2 + s_{b_m}^2}$$

The ratio

$$t = \frac{b_h - b_m}{s_{b_h - b_m}} = \frac{b_h - b_m}{\sqrt{s_{b_h}^2 + s_{b_m}^2}}$$

is distributed as t on N + N – 4 *df*.

The s_{b_i} can then be estimated by $s_{b_i} = \frac{s_{Y_i.X}}{s_X\sqrt{N-1}}$, $i = h \; or \; m$ (h denotes *human*, m denotes *model*).

Therefore,

$$s_{b_h-b_m} = \sqrt{\frac{s^2_{Y_h.X}}{s^2_X(N-1)} + \frac{s^2_{Y_m.X}}{s^2_X(N-1)}}$$

In our case, $s_{b_h-b_m} = \sqrt{\frac{8198}{20(14)} + \frac{12}{20(14)}} = 5.42.$

We now solve for t as follows.

$$t = \frac{b_h - b_m}{s_{b_h-b_m}} = \frac{-5.66 - (-4.73)}{5.42} = -0.17$$

with the degree of freedom $df = \mathrm{N} + \mathrm{N} - 4 = 30 - 4 = 26$.

For $\alpha = 0.05$ (two-tailed), the critical $t(26) = \pm 2.056$. Since the obtained t-score –0.17 lies between the critical t-scores –2.056 and 2.056, I would fail to reject H_0 and would therefore conclude that *I have no reason to doubt that the mean non-Fitts time decreases as a function of practice sessions at the same rate for the model as for the human*[3].

3.6.4.2 Achieved effect size of the model

I test the difference between the population mean NFTs of session 1 and session 15 through *matched sample* t-test. I do this to compute the estimated effect size $\hat{d}$ between session 1 and session 15. Table 3.4 shows the test results.

[3] I have reported the results here following the style of Howell (2007, p. 259).

Table 3.4 Results of testing the difference between the model population mean NFTs of session 1 and session 15.

Sessions pair compared	Difference of sample mean NFTs (ms)	SD of Difference of sample NFTs (ms)	t	p-value (two-tailed)	Effect Size, $\hat{d}$	Power
1st and 15th	74	142	11.62	< .05	0.52	> 0.99

$\alpha = 0.05$ (two-tailed). Sample size per session N = 500. df = 499. Critical t(499) = ±1.96. Non-centrality parameter δ can be computed as $\delta = \hat{d} * \sqrt{N}$ for matched samples where N is the sample size (Howell, 2007, p. 224). <.05 implies that p-value (two-tailed) for the pair of sessions is less than .05. >0.99 implies that power of the t-test for comparing the population means of the pair of sessions is greater than 0.99.

As found from Table 3.4, a *matched sample* t-test of the difference between the model population mean NFTs of 1st and 15th sessions produces a statistically significant result: $t(499) = 11.62$, $p < .05$, given $\alpha = 0.05$ (two-tailed), critical $t(499) = \pm 1.96$. The *effect size* $\hat{d} = 0.52$ shows that the two sessions differed by nearly 0.52 standard deviations of the difference of sample NFTs. This effect size of 0.52 is close to the effect size of interest 0.56 noted earlier in section 3.6.2 that we wanted our model to achieve.

3.6.5 Model based predictions

I test the difference between the model population mean NFTs of session 1 and session 160 through *matched sample* t-test. I do this to reveal that, on average, the non-Fitts time (NFT) *did* decrease over the course of practice, implying performance improvement. Table 3.5 shows the test results.

Table 3.5 Results of testing the difference between the model population mean NFTs of session 1 and session 160.

Sessions pair compared	Mean of Difference of sample NFTs (ms)	SD of Difference of sample NFTs (ms)	t	p-value (two-tailed)	Effect Size, $\hat{d}$	Power
1st and 160th	941	81	259.71	< .05	11.62	> 0.99

α = 0.05 (two-tailed). Sample size per session N = 500. df = 499. Critical t(499) = ±1.96. Non-centrality parameter δ can be computed as $\delta = \hat{d} * \sqrt{N}$ for matched samples where N is the sample size (Howell, 2007, p. 224). <.05 implies that p-value (two-tailed) for a pair of sessions is less than .05. >0.99 implies that power of the *t*-test for comparing the population means of a pair of sessions is greater than 0.99.

As found from Table 3.5, a *matched sample* t-test of the difference between the model population mean NFTs of 1st and 160th sessions produces a statistically significant result: $t(499) = 259.71$, $p < .05$, given $\alpha = 0.05$ (two-tailed), critical $t(499) = \pm1.96$. The 95% Confidence Interval on the mean of difference of the population NFT of the 1st session and the population NFT of the 160th session is (934 ms, 948 ms).

Recall Accuracy curve

The *recall accuracy*, *RA* (i.e. the number of successful memory retrievals in a session divided by the total number of retrieval attempts at that session) influences the *visual exploration time*. In case of my model, the total number of retrieval attempts in every session is 5 since a group of 5 letters are copied per session. Across the three stages of learning, *RA* controls the shift of behaviour from *visual search* to *choice*. Figure 3.7 shows the plot of *RA* against 160 sessions.

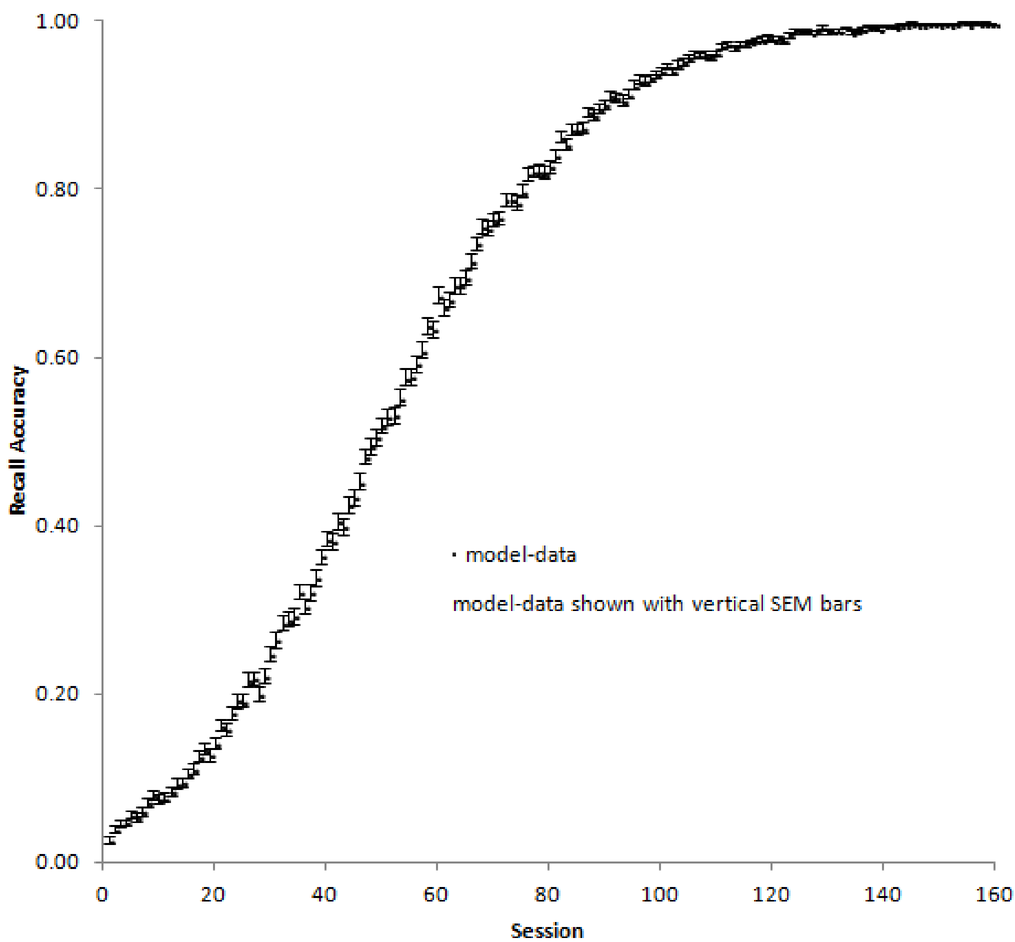

Figure 3.7 Modelled mean recall accuracy *RA* over 500 runs across 160 sessions. SEM bars are shown on the model data points.

I test the difference between the population mean *RA*s of session 1 and session 160 through *matched sample* t-test. I do this to reveal that, on average, the *RA did* increase for the pair over the course of practice. Table 3.6 shows the result of comparison.

Table 3.6 Results of testing the difference between the model population mean *RA*s of session 1 and session 160.

Sessions pair compared	Mean of Difference of sample *RA*s (ms)	SD of Difference of sample *RA*s (ms)	t	p-value (two-tailed)	Effect Size, $\hat{d}$	Power
160th and 1st	0.97	0.078	279.76	< .05	12.44	> 0.99

$\alpha = 0.05$ (two-tailed). Sample size per session N = 500. df = 499. Critical t(499) = ±1.96. Non-centrality parameter δ can be computed as $\delta = \hat{d} * \sqrt{N}$ for matched samples where N is the sample size (Howell, 2007, p. 224). <.05 implies that p-value (two-tailed) for a pair of sessions is less than .05. >0.99 implies that power of the *t*-test for comparing the population means of a pair of sessions is greater than 0.99.

As found from Table 3.6, a *matched sample* t-test of the difference between the model population mean *RA*s of 1st and 160th sessions produces a statistically significant result: $t(499) = 279.76$, $p < .05$, given $\alpha = 0.05$ (two-tailed), critical $t(499) = \pm 1.96$. The 95% Confidence Interval on the mean of difference of the population RA of the 160th session and the population RA of the 1st session is (0.964, 0.977).

3.6.6 Which stage of learning does a human data point belong to?

Given a single measured data point reflecting a learner's performance, I will try to predict which stage of learning the learner belongs to. In this regard, I will use two figures: One is the Figure 1.1, concluded by Kim, Ritter and Koubek (2013), that shows the shape of the learning curve depicting different stages of learning—Stage I (early stage), Stage II (intermediate stage) and Stage III (late stage). The other is Figure 3.5 which is the learning curve predicted by my model for the text copying task on a traditional phone keypad of a Nokia 5190 cell phone. These two curves are roughly similar in shape.

I now go back to the discussion of the single measured data point to identify where it belongs to (roughly) in the learning curve (Figure 3.5) of my model. Pavlovych and Stuerzlinger (2003) observed a text entry task of copying English sentences using a traditional phone keypad shown earlier in Figure 3.3. They reported a mean entry speed of 7.15 words per minute (wpm).

Assuming five letters per word, the *T*ask *C*ompletion *T*ime to enter a letter, *TCT*, would be *TCT* = (1 / 5) * (1 / wpm) * 1000 * 60, where *TCT* is in milliseconds. The assumption of five letters per word follows standard typists' definition of a word as five characters (MacKenzie & Soukoreff, 2002; p. 158). From the mean entry speed of 7.15 wpm observed by Pavlovych and Stuerzlinger (2003) noted above, I obtain the mean time to enter a letter to be 1678 ms (TCT = (1 / 5) * (1 / wpm) * 1000 * 60 = (1 / 5) * (1 / 7.15) * 1000 * 60 = 1678).

Earlier, in the Literature Review chapter, section 2.9, I had predicted the mean Fitts time of 302 ms for one-handed thumb entry on the traditional phone keypad. Assuming that the participants in the study of Pavlovych and Stuerzlinger (2003) used either their left or right thumb to enter text, I subtract the mean Fitts time of 302 ms from the task completion time of 1678 ms to obtain the mean non-Fitts time that would have been observed by Pavlovych and Stuerzlinger (2003). That observed mean non-Fitts time would have been 1376 ms (Non-Fitts time = *T*ask *C*ompletion *T*ime – Fitts time = 1678 – 302 = 1376).

Figure 3.5 is the non-Fitts time curve of 160 sessions obtained from my model. Figure 1.1 is the generic learning curve from Kim, Ritter and Koubek (2013), showing the three stages of learning. Comparing the shape of these two curves, and then eyeballing the non-Fitts time curve in Figure 3.5, I find that the mean non-Fitts time of 1376 ms appears to be occurring at Stage II of learning. This is in agreement with Pavlovych and Stuerzlinger (2004) who had concluded the human mean entry speed of 7.15 wpm (observed by them in 2003) to be a non-expert performance.

Thus, a learning curve obtained from my model could help in identifying which stage of learning a learner's expertise lies in. This may save training time and cost and help allocating training resources appropriately. My model therefore could become a useful complement to the experimental evaluation.

3.7 Discussion and Conclusions

I have proposed a model that predicts how the *visual exploration time* to find a pre-cued symbol on a layout affects the *non-Fitts time*. The prediction demonstrates that as the *recall accuracy* for a symbol increases with practice, the user gradually changes her exploration strategy from *visual search* of the symbol towards *choice* of the symbol location (button) from among the known alternative buttons.

I model the *visual exploration time* in terms of a mathematical equation and avoid implementing any custom simulation model for visual search. Although such a custom search model may provide a richer description of visual search strategies, my mathematical equation is less complex and more straightforward to apply.

I have demonstrated the effect of *visual exploration* on *non-Fitts time* by developing a simulation model for text copying task on a phone keypad layout. A similar effect can be demonstrated in other situations such as item acquisition on graphical layouts using the modelling concepts described in this chapter as follows: The VST should be provided; a simulation sub-model should be developed that will predict the NET and RA for each session; the parameters of the simulation sub-model should be tuned using the empirical data provided for the first few sessions; the VET for each session should be predicted using the *visual exploration time equation*, that utilizes the VST and the RA of the given session, as well as the CRT (computed using the number of buttons). Finally the *non-Fitts time* for each session can be predicted by adding the NET and VET of that session.

I have tested the novice part of my model's prediction against human data. The human data contained considerable oscillations. Although correlation between the human data and model data is a way to show the degree of match between them (Grant, 1962, as cited in Ritter et al., 2011), in the present case a direct correlation between the human and the model data makes little sense due to these oscillations. Therefore I tested the difference between the independent slopes of the regression lines of novice human data and novice model data (Yet, I do acknowledge that this is a weak way of testing a model against human data. Rather, the R^2 measure should be used to report the quality of fit between model data and human data, whenever possible, as Grant (1962) indicates). For $\alpha = 0.05$ (two-tailed), I concluded I have no reason to doubt that the mean non-Fitts time decreases as a function of practice sessions at the same rate for the model as for the human.

My model can help predict the expertise level of a learner. The prediction can be achieved from the learning curve generated using my model. Depending on the predicted stage of learning the learner is in, one can identify how many more sessions of practice would be necessary for the learner to achieve mastery over the task. Roughly knowing the expertise level a learner is in, may save training time, as one then can allocate training resources appropriately.

The limitations of this work are as follows.

My model does not account for the effect of potential errors that may be committed by entering unexpected characters while copying text. A modification of the current model to accommodate the effect of such errors is not a straightforward task. Future investigation is therefore warranted in this regard.

To model an expert user, I predict the *choice reaction time* for choosing a button containing the target letter to be copied. My model therefore becomes constrained by its dependence on the *choice reaction time* as follows: In a key-pressing task on a keyboard, Seibel (1963) had observed that the *choice reaction time* increased for 2 to approximately 8 alternatives, and showed trivial further increase no matter how many additional alternatives were added to the task. Thus, being dependent on the *choice reaction time*, my model becomes constrained by the limitation of a maximum of 8 alternative buttons. However, if one were to develop a similar model for graphical layouts, a maximum of 12 alternative buttons may be supported to model the choice reaction time (Cockburn et al., 2007a; Ahlstrom et al., 2010; Cockburn & Gutwin 2010).

I use Hick's Law (Hick, 1952) to model the choice reaction time. Hick's Law postulates that choice reaction time increases log-linearly with the number of choice alternatives. Having used Hick's Law, my second model is constrained by its limitations as follows: Kveraga, Boucher and Hughes (2002, as cited in Bogacz, Usher, Zhang & McClelland, 2007, p. 1669) observed that in tasks involving saccades to visual targets where one of the alternatives receives much more support than all the others, Hick's Law is violated and the choice reaction time does not depend on the number of alternatives. Besides, Lawrence, St. John, Abrams and Snyder (2008) observed that for saccadic eye movements, the choice time may decrease as number of alternatives increases, in contrast to predictions based on Hick's Law.

My simulation sub-model is limited in that it avoids repeated key presses required to arrive at a letter on a traditional phone keypad (see Figure 3.3 for the layout). For example, to copy the character sequence `cei`, the user needed to press the key containing `c` only once instead of pressing it thrice (refer to Figure 3.3 for the location of `c`), the key containing `e` only once instead of pressing it twice, and the key containing `i` only once instead of pressing it thrice. I do this to stay compatible with the specific user study of Pavlovych and Stuerzlinger (2004) that I validate my model against.

My model in this chapter is restricted to a layout of items at a single level[4] (i.e. a non-hierarchical layout). However, the model can be adapted to multi-level layouts (i.e. hierarchical layouts) as well. Let us consider an item acquisition task in the hierarchy starting from the root level of the hierarchy. In that case, the total non-Fitts time to acquire an item at a given level of the hierarchy at a given practice session could be predicted by summing the predicted non-Fitts times at that level and all the prior levels at that session.

I have tested the novice part of my model's prediction against human data. However, the *progression* along the learning curve from novice to expert level is yet to be validated. With the ubiquity of cell phones, such a validation seems difficult due to the lack of novice subjects, the boredom of the subjects associated with time consuming longitudinal experiments, and the financial burden in form of remuneration to be paid to the subjects.

To compute the VET for a letter, I need its VST. The value of VST was derived from the human data observed on a Nokia 5190 phone keypad (Figure 3.3). This is a traditional layout that a typical phone user is very familiar with. If a different phone keypad layout is used, a different value may be necessary. I suggest that a study be undertaken on other phone keypad layouts to investigate this possibility.

[4] An example of a non-hierarchical layout is a computer keyboard whereas an example of a hierarchical layout is a cascaded menu in applications like Adobe Acrobat or Firefox internet browser.

Chapter 4
A Closed-Form Model to Compare Effort to Learn Layouts

4.1 Introduction

The work presented in this chapter is related to the peer-reviewed material of Das and Stuerzlinger (2010, 2012, 2013).

Some layouts are easier to learn than others (Ehret, 2002; Cockburn et al, 2007b). The layouts that are easier to learn have also been observed to be easily relearnable implying that they can be learned again easily after forgetting (Kim & Ritter, in press). A certain amount of *effort* needs to be expended to *learn* a layout (Gray & Fu, 2004). I term this effort *mental effort*.

The level of *mental effort* required to learn a layout is influenced by several factors such as (i) the *effort* to be expended in search to obtain the target information (Casner & Larkin, 1989, as cited in Ritter & Larkin, 1994). This refers to the overall *effort* expended in activities such as planning, search strategy, spatial judgement, evaluation of items, and the actions carried out during the search for the target information; (ii) the amount of *knowledge* about the layout available (Kotovsky & Simon, 1990).

Ehret (2002) and Cockburn et al. (2007b) observed that poorer the *label representativeness* of items on a layout, the harder it is to learn the layout. Poorer label representativeness restricts the amount of knowledge and strategies. This in turn increases the *effort* to search for the target item.

Goal

The goal of this chapter is to develop a closed-form model that helps to quantitatively compare the level of *mental effort* expended to learn layouts in different *information access conditions* (henceforth called *access condition*). An access condition of a layout reflects a particular level of difficulty in acquiring the items on the layout. In this chapter, an access condition is represented by the *label representativeness* of the layout.

Motivation

Proactive Interference refers to the difficulty in recalling a target item caused by prior encoding of non-target items (*distractors*). Underwood (1957) observed that lower the number of *distractors*, the lower is the *proactive interference*. Proactive interference causes loss of memory activation. People exert *mental effort* to mitigate the effect of such interference.

Learning is influenced by multiple factors. Some of them are *practice*, *decay*, *interference* and *mental effort*. In human memory research, Rowe et al. (2008) suggested that *practice* positively influences spatial learning while *proactive interference* impacts it negatively. On the other hand, Altmann and Schunn, (2002) concluded that not only proactive interference but also *decay*, i.e., loss of memory activation with passage of time, is responsible for forgetting.

Taking into account the mutually constraining effects of *practice*, *mental effort*, *proactive interference* and *decay*, an integrated, yet simple and easily applicable performance model seems viable that reflects the effect of these phenomena on spatial learning.

Following this idea, I propose a closed-form model of spatial learning that combines the effect of *practice* in terms of *age of practice*, the effect of *decay* in terms of a *numeric constant*, the effect of *proactive interference* in terms of *Distractor Cost*—number of distractors visually encoded while searching for a pre-cued target item, and *effort factor*—a model parameter that quantifies the *mental effort*. All these effects are expressed in a single equation of memory activation. To achieve this, I adapt an existing memory activation model of ACT-R cognitive theory developed by Anderson et al. (2004).

The advantage of my model is that it can be used to *quantitatively* compare the level of *mental effort* expended to learn layouts in different *label representativeness*.

I consider the effect of the number of distractors on the proactive interference. However, I do not consider the effect of visual similarity between the distractors and the target on proactive interference.

I implement my closed-form model in a spreadsheet and validate it against two sets of empirical data previously collected by other researchers. My model is a deterministic model. It does not account for activation noise.

4.2 The Model

4.2.1 Motivation

To develop my closed-form model, I exploit the equations of the ACT-R declarative memory discussed in Section 2.8. I stay within the framework of the ACT-R *reaction time equation* of declarative memory, $RT_{n+1} = I + Fe^{(-f*A_{n+1})}$, $n \geq 1$. In this equation, RT_{n+1} is the reaction time of the $(n+1)^{th}$ practice. RT_{n+1} depends on the activation A_{n+1} of the item being practiced. A_{n+1} is the activation of the item during $(n+1)^{th}$ practice. $A_{n+1} = B_n + O_{n+1}$ where B_n is the base-level activation of the item after n practices have been completed—B_n is computed just before the $(n+1)^{th}$ practice happens. B_n is given by the base-level activation equation $B_n = ln\left(\sum_{j=1}^{n} t_j^{-d}\right)$, $n \geq 1$. O_{n+1} denotes the optional terms. The optional terms are accounted for when a practice is in progress. Thus, O_{n+1} is accounted for when the $(n+1)^{th}$ practice is in progress. I modify the base-level activation equation to model the effect of the *proactive interference* and the *mental effort*.

In the *reaction time equation* $RT_{n+1} = I + Fe^{(-f*A_{n+1})}$, I is an intercept time reflecting the fixed time cost of perceptual (visual) encoding and motor response (Anderson et al., 2004, p. 1043). F is the latency factor, and maps the activation to time. f is the latency exponent. The *reaction time* does not depend on the estimation of the parameters I and F. The effect of I and F is only to scale the critical quantity $e^{(-f*A_{n+1})}$ onto the range of the latencies.

Two previous work that motivate my model development in this chapter are Anderson (1983) and Stewart and West (2007). Anderson (1983, p. 277) had used a scaling factor as a coefficient of the age of a practice event to reflect the strength of that event. Stewart and West (2007, p. 235) conjectured that when the trace of an item is inserted into memory, it also strengthens the activation of related traces already present in the memory by certain amount. To reflect this increment in strength, Stewart and West suggested a scaling factor for the t_j^{-d} terms in the base-level activation equation.

The development of my model is also influenced by Pavlik, Presson and Koedinger (2007). They used the ACT-R Reaction Time equation of declarative memory $RT_{n+1} = I + Fe^{(-f*A_{n+1})}$, $n \geq 1$ to analytically compare the *learning difference* quantitatively between a *study practice* session and a *test practice* session in a paired-associate memory task. The *study practice* session involved visual encoding of two words in a pair that were presented on a computer screen. It did not involve any recall. A *test practice* session involved recalling the second member of a pair of words when the first member was presented on a computer screen. Pavlik et al. formulated the chunk *activation* A_{n+1} so that it consisted of a modified form of the base-level activation equation. They replaced the decay constant d with d_j to account for the *spacing effect* and included a parameter b_j as a coefficient of $t_j^{-d_j}$. Their logarithmic term in their modified base-level equation therefore had the form $ln\left(\sum_{j=1}^{n} b_j t_j^{-d_j}\right)$.

The parameter b_j was to compare the *learning difference* between *practice sessions*. The value of the parameter came out to be different for the practice session involving recall (i.e. test practice) in comparison to the practice session that did not involve recall (i.e. study practice).

The closed-form model of Pavlik et al. (2007) discussed above motivates me to use a modified form of the base-level activation equation B_n to account for the effect of *proactive interference* and to reflect the difference in the *mental effort* required to *learn* different layouts; given that the layouts differ in terms of their *access condition* (e.g. label representativeness of items). The approach of Pavlik et al. also motivates me to model the *mean task completion time* per item using the ACT-R *reaction time equation* of declarative memory $RT_{n+1} = I + Fe^{(-f*A_{n+1})}$.

Finally, another important work that motivates my modelling of *mental effort* is the *soft constraints hypothesis* of Gray and associates (Fu & Gray, 2001, 2004; Gray, Sims, Fu, & Schoelles, 2006). The hypothesis proposes that the mixture of effort—*perceptual-motor search effort*, *perceptual-motor access effort*, *memory encoding effort*, and *memory retrieval effort*—is allocated for interactive behaviour in a way that the *least-effort* path of executing the spatial task at hand gets implicitly chosen (Fu & Gray, 2001, 2004). As the acquisition of information from the environment becomes *harder*, people get motivated to choose the least-effort option of retrieving the information from memory, even if the memory retrieval is imperfect. Conversely, when acquisition of information from the environment becomes *easier*, people get motivated to choose the least-effort option of accessing information from the environment. Specifically, the work of Gray and associates motivates me to develop my model assumptions.

4.2.2 Assumptions in the model

The main equation of my model is the ACT-R *reaction time equation* of declarative memory $RT_{n+1} = I + Fe^{(-f*A_{n+1})}, n \geq 1$. In this equation $A_{n+1} = B_n + O_{n+1}$ where O_{n+1} denotes the optional terms and $B_n = ln\left(\sum_{j=1}^{n} t_j^{-d}\right)$. To keep my model simple, I ignore the optional terms. As I have noted earlier in the chapter on Literature Review, Section 2.8, ignoring the optional terms for simplifying model representation is not an exception. It follows previous work of Altmann and Schunn (2002) on modelling proactive interference, and Cochran, Lee and Chown (2006) for modelling the arousal effect. Therefore, from now onwards, I represent the ACT-R *reaction time equation* as $RT_{n+1} = I + Fe^{(-f*B_n)}, n \geq 1$, by replacing the term A_{n+1} by B_n for my modelling purposes. I will modify B_n to model the effect of *proactive interference* and the *mental effort* in the next subsections.

I intend to validate my model stand-alone, without merging it in the ACT-R simulation framework. I intend to do so to bypass the expertise required in the merger. Doing so, I forgo a richer, albeit complex, description of behaviour. On the other hand, I intend to develop a model that is simple and straightforward to apply. To fulfil my intention, I need to simplify the description and analysis of my model. In this regard, I make certain assumptions. They are as follows.

In the process of finding a pre-cued target item on a layout, (i) I assume that a subject is *unfamiliar* with the layout before the start of the first practice session.

Therefore no item is recalled in the *first* practice session. (ii) I assume that from the *second* practice session onwards, the *reaction time* to find the target item at a given session is affected by the number of distractors encountered in the *previous* sessions.

As I have mentioned in the chapter on Literature Review, section 2.6, Fu and Gray (2001, 2004) had conjectured the existence of two combinations of effort components that goes into the learning of a visuo-spatial task. These combinations are (i) *the perceptual-motor access effort + the related memory encoding effort + the related memory retrieval effort*, (ii) *the perceptual-motor search effort + the related memory encoding effort + the related memory retrieval effort.*

Fu and Gray (2001, 2004) further conjectured that the first aforementioned combination—*the perceptual-motor access effort + the related memory encoding effort + the related memory retrieval effort*—is expended predominantly in the *expert* phase of the learning curve. Moreover, Fu and Gray were able to successfully interpret these effort components as the effort analogue of the terms in the default ACT-R reaction time equation $RT_{n+1} = I + Fe^{(-f*B_n)}$ (Fu & Gray, 2001, p. 112).

Fu and Gray (2001) however suggested that the second aforementioned combination—*the perceptual-motor search effort + the related memory encoding effort + the related memory retrieval effort*—is expended predominantly in the *non-expert* phase of the learning curve (Fu & Gray, 2001, p. 112; Fu & Gray 2004; p. 366). They *ignored* this second combination since their main interest laid in modelling the *expert* phase of the learning curve.

Unlike Fu and Gray (2001, 2004), I intend to account for the second combination of effort—*the perceptual-motor search effort + the related memory encoding effort + the related memory retrieval effort*—to reflect the effort that goes predominantly in the *non-expert* phase of learning. For my model analysis, I refer to this second combination as *mental effort*. The *mental effort* will subsequently be reflected by a new model parameter *effort factor* that I will introduce later in this chapter.

Next, I propose my extension to the base-level activation equation. I introduce the extension to account for the effect of *proactive interference* and the *mental effort*. I do so largely by adapting existing cognitive constructs rather than developing new ones.

4.2.3 Modelling the Proactive Interference

Proactive Interference (PI) refers to the difficulty in recalling a target item caused by prior encoding of non-target items (*distractors*). In the domain of verbal learning, Underwood (1957) holds the *number* of distractors to be responsible for *proactive interference*. The lower the *number* of distractors is, the lower is the PI. Similar observations were made by Elmes (1988, p. 672) in the domain of spatial learning.

To account for PI in my model, I replace the decay constant d of the base-level activation equation $B_n = ln\left(\sum_{j=1}^{n} t_j^{-d}\right)$ with a new function described next. I assume that the effect of PI in a given session is due to the *number* of distractors visually encoded in the previous sessions.

The new function will consist of a *constant term* and a *functional term*. The *constant term* will model the *decay*—the loss of memory strength with the passage of time—as in classic ACT-R. The *functional term* will model the loss of memory strength due to *proactive interference*. My proposal for modelling the combined effect of *decay* and *interference* on memory activation is in line with the observations of Altmann and Schunn (2002), which indicts both decay and proactive interference for forgetting.

The *functional term* I propose is a function of the *Distractor Cost*—the number of distractors that get visually encoded prior to encoding a pre-cued target item when one tries to find the said target item on a layout in a practice session. The *Distractor Cost* contributes to my measure for the *proactive interference* effect: the *lower* the number of distractors is, the *lower* the *loss* of activation of the target item should be. Consequently, the *reaction time* to find the target item in the next practice session will be lowered. This will show an improvement in search-and-selection performance during exploration of the layout in question. My hypothesis is grounded in the primary research result of Underwood (1957) on proactive interference. His research identified the effect that the number of previously learned items has on the recall of the target item: the *lower* the number of previously learned items is, the *lower* is the forgetting effect and therefore the *lower* is the retrieval latency for the target item.

The new function d_j that replaces the decay constant d of $B_n = ln\left(\sum_{j=1}^{n} t_j^{-d}\right)$ is as follows.

$$d_j = h + 0.5 * X_j/N \qquad \textbf{Decay Rate Equation}$$

I call d_j the *decay rate*. The decay rate d_j for an item reflects how quickly the memory strength of the item diminishes once j practices for the said item have been completed.

In the decay rate equation above, h is the *constant term* and $(0.5 * X_j/N)$ is the *functional term*. h represents the time-based decay constant; it models the loss of memory strength with the passage of time. The term $(0.5 * X_j/N)$ models the loss of memory strength due to *proactive interference*.

In the term $(0.5 * X_j/N)$, N is the total number of items on the layout. X_j is the *Distractor Cost* at j^{th} practice, i.e. the mean number of distractors that have been visually encoded at j^{th} practice. j is greater than or equal to 1. When X_j is 0, i.e., when the user is able to complete the task by direct recall or does not encounter any distractor at j^{th} practice, the decay rate equation degenerates to $d_j = h$. This implies that, in the absence of the impact of distractors, loss of memory strength occurs only with the passage of time, as in classic ACT-R.

The product term $0.5 * X_j/N$ transforms the number of distractors X_j to a *decay* value. The ratio X_j/N ranges from 0 to 1. Consequently, the product term $0.5 * X_j/N$ yields a value in the interval, 0 to 0.5. The case of $0.5 * X_j/N = 0.5$ refers to a situation where the maximum possible number of distractors is encountered (i.e. when $X_j = N$), leading to the highest possible level of *proactive interference*. This, in turn, reduces the term to the maximum value of 0.5. On the other hand, $0.5 * X_j/N = 0$ implies an absence of impact from distractors, and therefore no *proactive interference*. This occurs when the user is able to complete the task by direct recall or when the user has not encountered any distractor at j^{th} practice. I choose 0.5 to be the upper bound of the product term $0.5 * X_j/N$. Although my choice of 0.5 is an ad-hoc one, yet the values of 0.5 or 0.6 have been used earlier for the decay constant d in different applications (Anderson et al., 2004, p. 1042; Halverson et al., 2010, p. 83).

My rationale behind replacing the decay constant d with a mathematical function d_j is motivated by the mathematical constructs for decay rate by Pavlik and Anderson (2005) and Pavlik et al. (2007) for the spacing effect, and that of Cochran, Lee and Chown (2006) for the arousal effect. Each of these works use decay rate functions instead of a decay constant for their respective memory models. All replace the decay constant with decay rate function in the base-level activation equation. I assume my decay rate function to be linear.

4.2.4 Modelling the Mental Effort

Fu and Gray (2001, 2004) conjecture that combinations of four effort components—*perceptual-motor search effort*, *perceptual-motor access effort*, *memory encoding effort*, and *memory retrieval effort*—get expended to select an item on an user interface. They suggest that a particular combination—*the perceptual-motor search effort* + *the related memory encoding effort* + *the related memory retrieval effort*—gets expended predominantly in the *non-expert* phase of the learning curve (Fu & Gray, 2001, p. 112; Fu & Gray, 2004, p. 366). Earlier, I referred to this combination as *mental effort*.

To account for *mental effort* in my model, I introduce a new parameter k as a coefficient of the t_j term in the base-level activation equation $B_n = ln\left(\sum_{j=1}^{n} t_j^{-d}\right)$. The introduction of this parameter is motivated by Anderson (1983), Stewart and West (2007) and Pavlik et al. (2007). Unlike the work of Pavlik et al. (2007), but similar to the works of Anderson (1983, p. 277) and Stewart et al. (2007, p. 235), the value of this parameter is to stay the same across all *practice sessions* for a given *access condition*. It may however differ across different *access conditions*. I call the new parameter k the *effort factor*. I hypothesize to use k for comparing access conditions among layouts.

I describe my modified base-level activation equation next. It accounts for both *proactive interference* and *mental effort*.

4.2.5 Modified Base-Level Activation Equation

With the *decay rate* d_j and the *effort factor* k conceptualized, I modify the base-level activation equation to

$$B'_n = ln\left(k \sum_{j=1}^{n} t_j^{-d_j}\right) \quad \textbf{Modified Base-Level Activation Equation}$$

The modified base-level activation equation B'_n above is obtained by including two new elements d_j and k to the original base-level activation equation. I explain the new elements in more detail below.

The element d_j is the *decay rate equation*. I introduced d_j in detail earlier. It consists of the sum of two terms—one representing the traditional *time-based decay constant* and the other representing the *loss of activation due to proactive interference*.

The element k in the equation is the aforementioned *effort factor* parameter. Later in this chapter, I explain k in the context of learning layouts that differ in terms of *label representativeness* (access condition) of their items.

4.2.6 Modified ACT-R Reaction Time Equation

Finally, the closed-form model is the modified ACT-R *Reaction Time Equation* given by

$$RT'_{n+1} = I + Fe^{(-f * B'_n)} \qquad \textbf{Modified Reaction Time Equation}$$

where $B'_n = ln\left(k \sum_{j=1}^{n} t_j^{-d_j}\right)$ is the *Modified Base-Level Activation Equation*; n is number of practice sessions completed so far, $n \geq 1$; j refers to the j^{th} practice session; t_j is the age of the j-th practice; k is the effort factor; $d_j = h + 0.5 * X_j/N$ is the *Decay Rate Equation*; h is the time-based decay constant; X_j is the mean number of distractors encountered at j^{th} practice session; N is the number of items on a layout under scrutiny; F is the latency factor; f is the latency exponent; I is the fixed time cost of visual encoding and motor response.

In the *Modified Reaction Time Equation*, F, f, h and k are the free parameters. The rest of the parameters are input parameters.

Given a set of layouts to be compared in terms of their *mental effort*, the free parameters F, *f* and *h* are to be held constant. The free parameter *k* is to vary across the layouts that differ in terms of access conditions.

I hypothesize a few properties related to *k* below. I assume that the layouts differ in terms of their access conditions. I further assume that the layouts are to be compared in terms of their modelled reaction time, as obtained from the modified ACT-R *reaction time equation*, $RT'_{n+1} = I + Fe^{(-f*B'_n)}$:

(i) The *effort factor k* quantifies the *mental effort* which refers to the combination—*the perceptual-motor search effort* + *the related memory encoding effort* + *the related memory retrieval effort*. This combination gets expended predominantly in the *non-expert* phase of the learning curve (Fu & Gray, 2001, p. 112; Fu & Gray, 2004, p. 366). This *effort* is consumed in finding a pre-cued target item on a layout.

(ii) A value of *k* corresponds to one particular layout, i.e., one particular access condition.

(iii) A *lower* value of *k* corresponds to a layout that would require *higher* mental effort, whereas a *higher* value of *k* corresponds to a layout that would require *lower* mental effort[5].

[5] A lower *k* would result from a higher *RT*. In contrast, a higher *k* would result from a lower *RT*. Given a practice session in the early stages of practice, a higher value of *RT* is typically evident for layouts with *higher* access cost whereas a lower value of *RT* is typically evident for layouts with *lower* access cost (as noticeable from the empirical data in Ehret (2002) and Cockburn et al. (2007b), for example).

4.3 Model Validation

I validate my model against two sets of empirical data previously collected by other researchers. Specifically, I use empirical data from two different experiments, one involving a *circle of buttons* on a computer screen (Ehret, 1999, 2002) and the other involving a *graphical keyboard* on a computer screen (Cockburn et al, 2007b). I obtained this empirical data by digitizing the screenshots of the graphs provided in Ehret (2002) and Cockburn et al. (2007b). I consider the *novice-to-expert* transition phase of the empirical data to validate my model.

The task I model here involve searching and selecting a pre-cued item on a structured layout of graphical buttons presented on a computer screen. Guided by Gray et al. (2006), I base the movement times on Fitts' law (MacKenzie, 1992), which predicts how long it takes a mouse cursor to move a given distance to an item of a given size.

To simplify the model development process, I predict the average movement time using Fitts' law to be 360 ms for the *circle of buttons* and 230 ms for the *graphical keyboard*. The reason I predict the movement time data using Fitts' law is due to the absence of such data in the reports of the empirical studies I validate against.

I now show how I arrived at the average movement times for the *circle of buttons* and the *graphical keyboard* mentioned above using Fitts' law (MacKenzie, 1992). I already explained Fitts' law earlier in the chapter on Literature Review, section 2.9. Fitts' law predicts the Movement Time MT it takes a pointing device to move a given distance to an item of a given size. I use MacKenzie's formulation of the law. It is expressed as follows.

$$MT = a + b * \log_2\left(\frac{A}{W} + 1\right) = a + b * ID$$ **Fitts' Law (MacKenzie's formulation)**

In the above equation of Fitts' Law, A is the amplitude of the movement (e.g. the distance between two keys on a keyboard—a source key where the movement begins from and a target key where the movement ends), and W is the width of the target item. The log term in the equation is called the *index of difficulty* ID. The reason behind the choice to use MacKenzie's formulation is to avoid a negative ID when the A/W ratio drops below 0.5.

I use the Fitts' law coefficients a = 0.05 sec; b = 0.10 sec/bit. The values of these coefficients are based on Card, English, and Burr (1978) and have been shown to provide a good fit for moving a mouse cursor around a computer screen (Gray et al., 2006).

It is to be noted that in the early and intermediate stages of learning, the movement time is only a small fraction of the total time needed to perform a target acquisition (Salthouse, 1986; John, 1996; Pavlovych & Stuerzlinger, 2004; Ahlstrom et al., 2010; Kim, Ritter & Koubek, 2013; Kim & Ritter, in press). Since I consider the *novice-to-expert* transition phase of the empirical data to validate my model, taking the *average* movement time for each of the interfaces, the *circle of buttons* and the *graphical keyboard*, is an acceptable compromise.

Average movement time prediction for circle of buttons

I digitize the screenshot of the *circle of buttons* reported by Ehret (1999). The screenshot is shown in Figure 4.1. I use Engauge Digitizer[6] version 4.1 for the digitization. Each button on the circumference of the circle is square shaped (Ehret, 1999). Using the digitizer, I set the width of a button as 1 unit. Then, in terms of the width of a button as one unit, I obtain an approximation of the maximum distance between the centers of two buttons, that is, the buttons that are at diametrically opposite locations and the minimum distance between the centers of two buttons, that is, the buttons that are adjacent to each other horizontally. These approximate distances are taken to be the maximum and minimum amplitudes respectively. In case of the *circle of buttons*, the maximum amplitude is 16.1 unit and the minimum amplitude is 3.32 unit.

[6] see http://digitizer.sourceforge.net

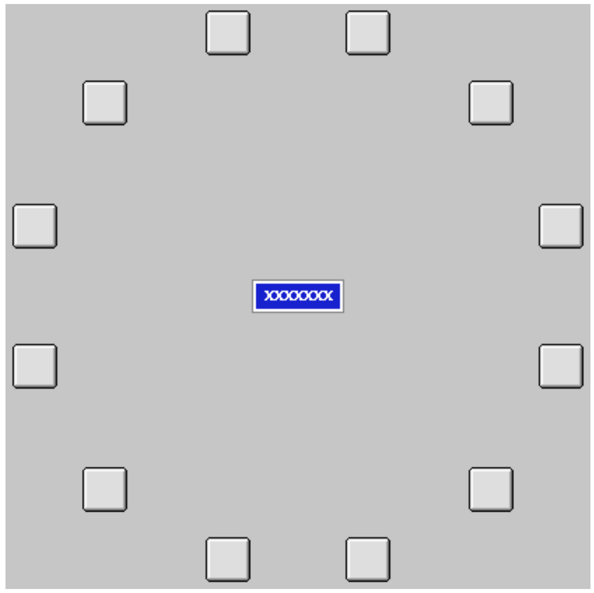

Figure 4.1 The *circle of buttons* layout. The figure is taken from Ehret (1999, p. 27, Figure 2d).

Next, using Fitts law (MacKenzie's formulation) equation above, I obtain the maximum $MT = a + b * \log_2(A/W + 1) = 0.05 + 0.10 * \log_2(16.1/1 + 1) \approx 0.4596$ and minimum $MT = a + b * \log_2(A/W + 1) = 0.05 + 0.10 * \log_2(3.32/1 + 1) \approx 0.2611$. Therefore the average is MT = (maximum MT + minimum MT) / 2 ≈ 0.360 sec. Thus the average movement time for the *circle of buttons* layout is predicted to be around 360 ms.

Average movement time prediction for graphical keyboard

I digitize the screenshot of the *graphical keyboard* reported by Cockburn et al. (2007b). The screenshot is shown in Figure 4.2. I again use the Engauge Digitizer version 4.1 for digitization. Each key on the graphical keyboard is assumed to be square shaped. Using the digitizer, I set the width of a key as 1 unit. Then, in terms of the width of a key as one unit, I obtain an approximation of the maximum distance between the centers of two keys, that is, the keys that are at maximum distance from one another (top-left key and bottom-right key) and the minimum distance between the centers of two key, that is, the buttons that are adjacent to each other horizontally (e.g. two adjacent keys at the top row). These approximate distances are taken to be the maximum and minimum amplitudes respectively. In this case, the maximum amplitude is 5.1 unit and the minimum amplitude is 1 unit.

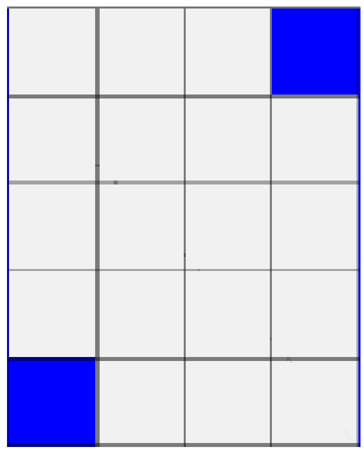

Figure 4.2 The *graphical keyboard* layout. The figure is adapted from Cockburn et al. (2007b, p. 1573, Figure 1).

Next, using Fitts law (MacKenzie's formulation) equation above, I obtain the maximum $MT = a + b * \log_2(A/W + 1) = 0.05 + 0.10 * \log_2(5.1/1 + 1) \approx 0.31$ and minimum $MT = a + b * \log_2(A/W + 1) = 0.05 + 0.10 * \log_2(1/1 + 1) \approx 0.15$. Therefore, the average is $\mathrm{MT} = (\text{maximum MT} + \text{minimum MT}) / 2 \approx 0.230$ sec. Thus the average movement time for the *graphical keyboard* layout is predicted to be around 230 ms.

4.3.1 Choosing the model parameter values

I now explain the rationale behind setting the model parameters to their relevant values. The time-based decay constant h in the decay rate equation was fixed at h = 0.058. I am motivated here by Pavlik and Anderson (2005, p. 572), who used it as a decay intercept, albeit in a different modelling context. Since the focus of my decay rate equation is to model the effect of proactive interference, I place greater emphasis on the role of distracting information. In this regard, I am motivated by the discourse of Altmann and Gray (2008, p. 628) who argue for the influential role of proactive interference in forgetting compared to the role of decay in the domain of distractor-affected learning. My choice of a very small value of the time-based decay constant is therefore appropriate.

The latency factor F in the reaction time equation is left at its default value of F = 1, as per classic ACT-R theory.

The latency exponent f in the modified reaction time equation $RT'_{n+1} = I + Fe^{(-f*B'_n)}$ is fixed to a constant value for a given set of layouts being compared. To compare the access conditions of Ehret's (2002) *circle of buttons* experiment, I determined f to be 0.68. To compare the access conditions of Cockburn et al.'s (2007b) *graphical keyboard* experiment, I determined f to be 0.26. During the process of finding a fixed value of f, I also find the values of k for the given access conditions of the layouts. Next, I discuss a *procedure* to find both the f and k values.

4.3.2 Procedure to determine the *f* value and the *k* values

Given a set of layouts to be ranked in terms of k, a value of f needs to be determined that should stay fixed across all the layouts.

For each layout, I set up an MS Excel spreadsheet to determine the R^2 and RMSE values of fitting the model reaction times against human reaction times across several sessions. The human reaction time here is an empirical reaction time to find a pre-cued target item at a given session. The model reaction time for a given session is computed using the modified reaction time equation $RT'_{n+1} = I + Fe^{(-f*B'_n)}$, with the number of sessions being $n \geq 1$. The first session is assumed to be the one that does not involve any recall of the item location.

Given the modified reaction time equation $RT'_{n+1} = I + Fe^{(-f*B'_n)}$, the range of f is $0 < f \leq 1$. I use the following steps to determine a fixed value of f and the value of k for each layout.

(i) A finite set F' of f values is chosen from the range $0 < f \leq 1$. Let $n(F')$ denote the cardinality of set F'.

(ii) A finite set K of k values is chosen such that $0 < k$. Let $n(K)$ denote the cardinality of set K.

(iii) For each layout, a $n(F') \times n(K)$ matrix of R^2 values of the data-fit is determined. Each element of the matrix is the R^2 value corresponding to a given pair $\langle f, k\rangle$[7]. From the $n(F') \times n(K)$ matrix, I could see that, for a given k, the effect of f on R^2 is notable. In contrast, for a given f, the effect of k on R^2 is negligible.

(iv) For a layout, if a cut-off minimum value of R^2 is *not* provided, then retain the original set of f values. Otherwise, use the cut-off minimum value of R^2 to determine a set of f values that meets or exceeds the said cut-off. Then repeat this step for all the layouts.

(v) Determine the set F'' of f values *common* across all the layouts that meet the minimum R^2 criterion for every layout.

(vi) I now use the set F'' determined in the previous step. For each layout, a $n(F'') \times n(K)$ matrix of RMSEs of the data-fit is determined. Each element of the matrix is the RMSE corresponding to a given pair $\langle f, k\rangle$[8].

(vii) For each layout, determine the minimum RMSE corresponding to each f value in the set F''.

(viii) For a layout, if a tolerable maximum RMSE is *not* provided, then retain the set F''. Otherwise, a tolerable maximum RMSE is provided—In that case, if the minimum RMSE obtained for a given f in the previous step is *not* less than or equal

[7] I have used the *What-If Analysis → Data Table* tool in MS Excel to determine the R^2 values of the $f \times k$ matrix.
[8] I have used the *What-If Analysis → Data Table* tool in MS Excel to determine the RMSE values of the $f \times k$ matrix.

to the tolerable maximum RMSE, then remove that f value from the set F''. This step results in an updated set F''' of f values.

(ix) Corresponding to each f value in F''', determine the sum of the minimum RMSEs across all the layouts.

(x) Determine the *minimum* among the sum of minimum RMSEs obtained in the previous step. Let this be called the grand minimum RMSE.

(xi) Finally, determine the value of f corresponding to the grand minimum RMSE obtained in the previous step. This is the *fixed f value* to be used for the set of layouts being compared.

(xii) Given a layout, determine the minimum RMSE corresponding to the *fixed f value* from the layout's $n(F''') \times n(K)$ matrix of RMSEs. Then corresponding to that minimum RMSE, determine the k value. This is the value of the *effort factor k* of the layout under scrutiny. Then, repeat this step to determine the k values for all the layouts.

I use the procedure outlined above to compute the values for f and k later.

4.3.3 Circle of Buttons (Ehret 1999, 2002)

Knowing an item's location can reduce a user's task time and errors. As the number of screen items increases, so does the utility of location knowledge. Ehret (2002) carried out an experiment that tests how the time to find a pre-cued item varies with the varying degree of *label representativeness* (i.e. access conditions) of items across layouts.

4.3.3.1 Ehret's task

Ehret used a *search and select* task. In a given instance of the task (i.e. in a trial), a participant was first presented with a particular colour in a *rectangle* positioned at the centre of the circle. At the start of the task, the colour in the rectangle was its background colour. There were 12 such colours—red, blue, light blue, green, light green, tan, brown, gray, orange, yellow, pink, and purple. The foreground of the rectangle contained seven white lower-case 'x' letters (e.g. see Figure 4.3). First, the subject would click on the central *rectangle* to display 12 *square* buttons arranged in a circle around it. Each square button was already mapped to one of the 12 colours. Next, the subject's goal was to find, point to, and click on a square button using a computer mouse that would make the seven white 'x' letters the same colour as that of rectangle's background colour, thereby making the rectangle appear solid. For a given subject, the square buttons and their respective labels appeared in the same locations along the circumference of the circle throughout the experiment (Ehret, 1999, p. 23). The contour and shape of every button was always visible across all conditions (Ehret, 1999, p. 27). To discourage errors, the computer would beep five times when participants clicked the wrong button, a dialog box would then appear, and the trial would have to be repeated (Ehret, 2002; p. 212). Figure 4.3, 4.4 and 4.5 shows the layout for the three access conditions *textual*, *arbitrary*, or *invisible* (later we explain the meaning of these access conditions) respectively.

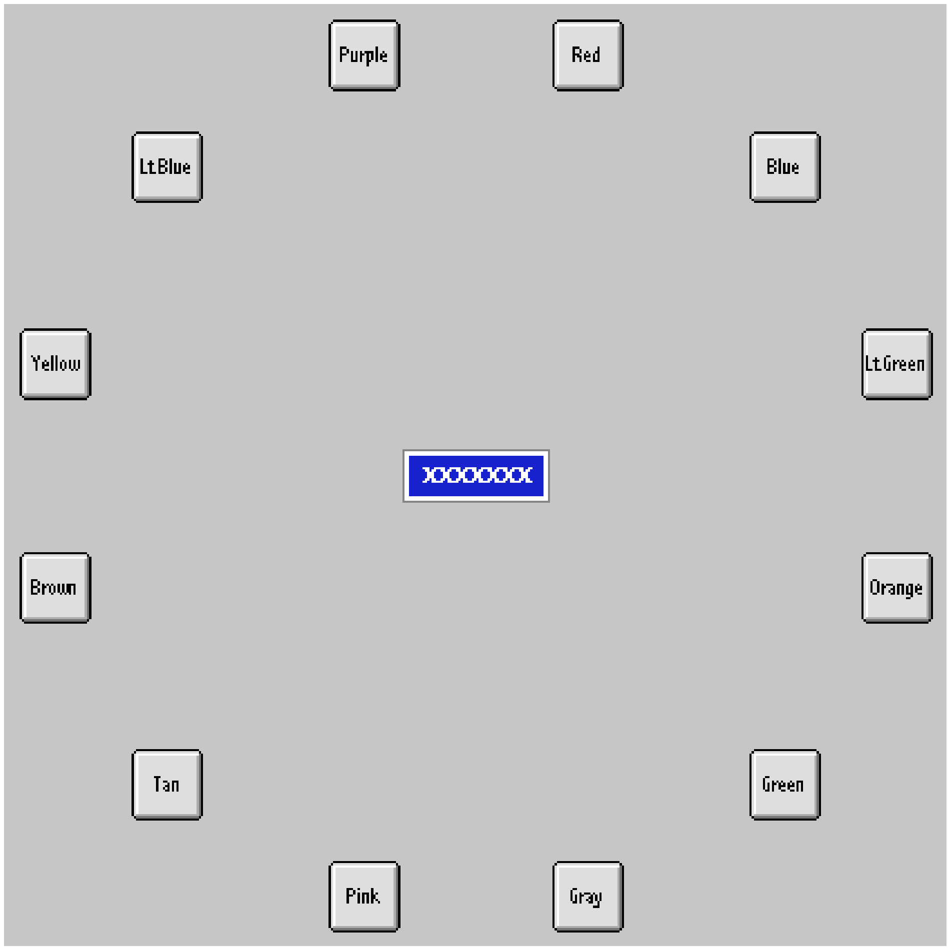

Figure 4.3 **The *circle of buttons* layout in the *textual* access condition. The button labels are 12 colour names in English. The figure is taken from Ehret (1999, p. 27, Figure 2b).**

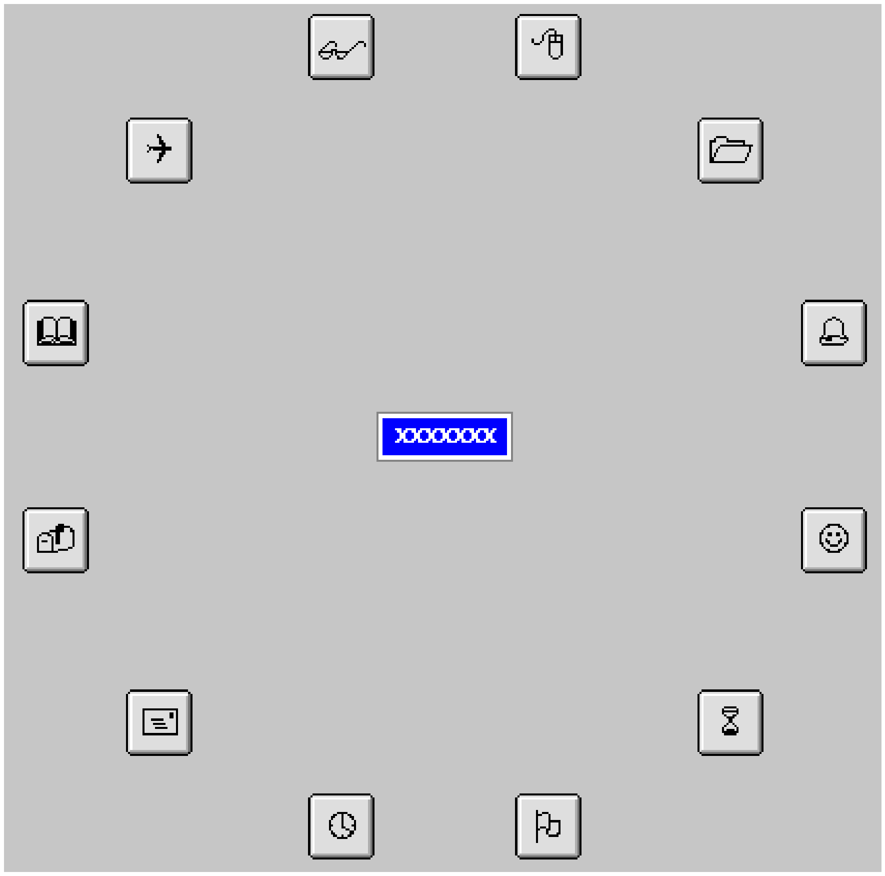

Figure 4.4 The *circle of buttons* layout in the *arbitrary* access condition. The button labels are icons bearing no particular relationship to any of the 12 colours. The figure is taken from Ehret (1999, p. 27, Figure 2c).

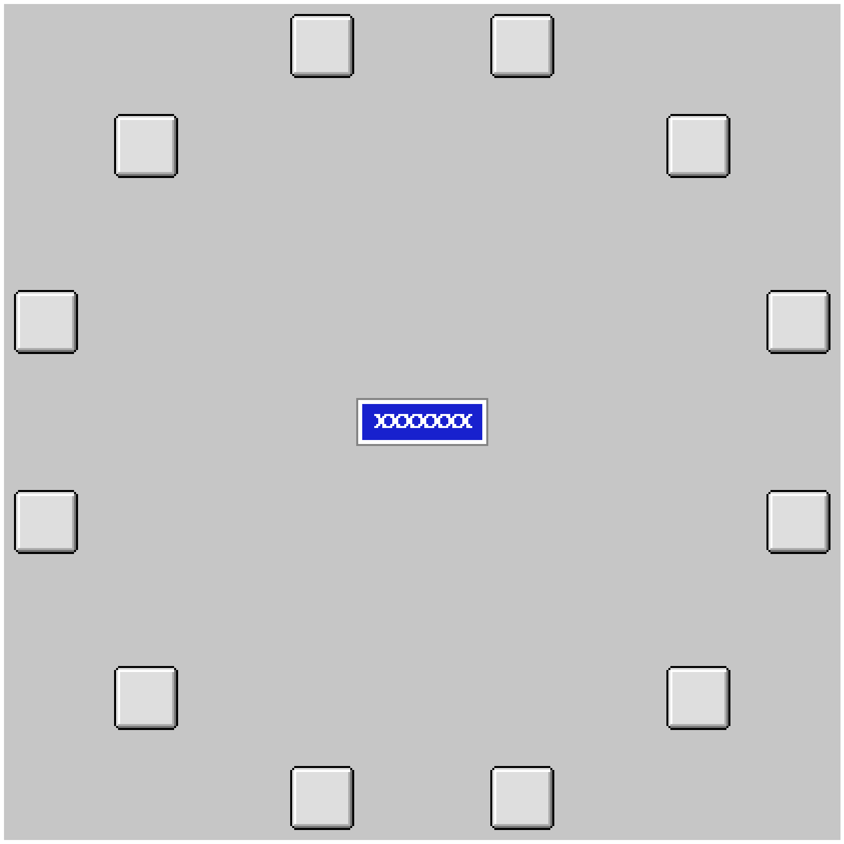

Figure 4.5 The *circle of buttons* layout in the *invisible* access condition. The buttons have no labels on them. The figure is taken from Ehret (1999, p. 27, Figure 2d).

4.3.3.2 Ehret's participants and design

There were sixteen subjects in the study. They were undergraduates participating in the study for course credit. They were randomly assigned to different access conditions *textual*, *arbitrary*, or *invisible*. They completed the task for 16 sessions of 12 trials each.

Subject's point-of-gaze data was measured as they performed the task. The point of gaze data was collected via an ASL 5000 eye-tracker. Two key measures were derived from the eye-tracking data: *Search cost*, operationalized as the mean number of *square* buttons attended in a given practice session, and *evaluation cost*, operationalized as the mean amount of time spent attending to a button in a given practice session. *Mean evaluation cost* per button for a given practice session was calculated as follows: The total time taken to complete all the trials by all the subjects in the session minus the total time spent in the central *rectangle* zone during all the trials by all the subjects in the session, and then divide the result of the subtraction by the total number of *square* buttons visited during all the trials by all the subjects in the session. The *mean evaluation cost* per button thus includes the mouse-cursor movement time as well as the mouse-click time.

4.3.3.3 Model Validation using human data from Ehret (2002)

In order to validate my model I extracted three data sets from Ehret's observations (Ehret, 2002; p. 214; Figure 2a and 3a). The three data sets correspond to three different access conditions. I did this by digitizing Figure 2a and 3a of Ehret's (2002) work. The data sets that I derived from the digitized information are the mean *search and select* time per item (i.e. mean *task completion time* per item) for the three access conditions. I next explain how I derived the empirical mean *task completion time* per item for a given session from the data of Figure 2a and 3a of Ehret (2002).

In his study, Ehret (2002, p. 214; Figure 2a and 3a) reported two empirical costs for a given practice session that I repeat here for the convenience of the reader. One is the *mean search cost* per target item for a given session (Ehret, 2002; p. 214; Figure 2a). It is the *mean number of square buttons* evaluated in the given session. It thus includes all the distractors and the target item in a given session. The other cost is the *mean evaluation cost* per item in a given session (Ehret, 2002; p. 214; Figure 3a). It is the mean amount of time spent attending to a square button in a given session. Ehret (2002) reported these two empirical costs for each of the three access conditions. For a given session, I arrive at an empirical *mean task completion time* per button in a session by multiplying the *mean search cost* with the *mean evaluation cost* corresponding to that session. I do this computation for every access condition.

The three data sets differed in the level of *representativeness* of labels (i.e. access condition) associated with the buttons.

The first data set corresponded to the *textual* access condition (see Figure 4.3). This data set was acquired while the subjects searched for a pre-cued colour in the buttons, each labelled with the name of a colour written in English. The aim was to have a *high* level of representativeness of the colours.

The second data set corresponded to the *arbitrary* access condition (see Figure 4.4). This data set was acquired while the subjects searched for a pre-cued colour in the buttons, each labelled with an arbitrary icon. The aim was to have a *lower* level of representativeness of the colours compared to the *textual* condition.

The third data set corresponded to the *invisible* access condition (see Figure 4.5). This data set was acquired while the subjects searched for a pre-cued colour among buttons with no labels on them. The aim was to have a *lower* level of representativeness of the colours compared to the *textual* as well as the *arbitrary* conditions.

In summary, each set of data consisted of *mean task completion times* per item (i.e. square button) for 16 sessions. Each set corresponded to one of the three levels of difficulty in accessing information: the *textual*, *arbitrary* or *invisible* access condition. Each condition represents a certain level of access cost, the *textual* condition featuring the lowest and the *invisible* condition the highest. *The total practice time was held constant across all access conditions*. For the *arbitrary* and *invisible* conditions a tooltip was provided for each button to aid the subject, if memory failed. Accessing the tooltip for a button revealed a small rectangle containing the colour associated with it. The cost of accessing this tip was a one-second delay between moving the mouse cursor to the button and the appearance of the tooltip.

My choice of data sets aligns with my modelling objective. I aim to model the combined effect of *Distractor Cost* (my surrogate of *proactive interference*) as well as *mental effort* on the *task completion time*, over the practice sessions. Since *Distractor Cost* is incurred due to distractors, it should not include the target item. Hence the *Distractor Cost* (i.e. the number of distractors) X_j at j^{th} session is one less than the *search cost* (i.e. total number of items examined) E_j at j^{th} session. Formally,

$$X_j = E_j - 1, \text{ where } j \geq 1 \qquad \textbf{Distractor Cost Equation}$$

Ehret's data shows that given an access condition, the *search cost* has a decreasing trend over the practice sessions implying that *proactive interference* (reflected by *Distractor Cost*) tends to decrease with practice.

Ehret's data further shows that in the early stages of practice, when the access condition increased from *textual* to *arbitrary* to *invisible* label conditions of buttons, so did the time to evaluate if a button currently under scrutiny is indeed the target or not, at any given session. This *evaluation cost* was observed to be the lowest for the *textual* label condition and highest for the *invisible* label condition. In other words, the layouts with higher access cost featured higher evaluation cost, implying also a *higher mental effort* to learn those layouts compared to the ones with lower access cost. In summary, a higher access cost condition would require higher *mental effort* compared to that required for a lower access cost condition.

Assumptions for the data fitting exercise

For the validation of my model against the data sets, I had to make a few assumptions, as certain information was not mentioned explicitly in the work of Ehret (2002). The assumptions are with respect to a given access condition, and with respect to a pre-cued target item to be found.

To find a pre-cued target item, (i) I assume that the first session occurs at time 0. (ii) I assume that the first session is equivalent to a *study practice* session implying that a subject searches a layout for the target item in the first session. *No item can be recalled in the first session since the subject is scanning the layout for the very first time in that session.* (iii) I assume that recall happens from the second session onwards. Recall at a given session is affected by the number of distractors encountered in the previous sessions. (iv) In the absence of any inter-session data (i.e. inter-trial data) in the study, I assume that the consecutive sessions were equally spaced.

Ehret (1999, p. 136) had expressed that 16 sessions took 10 minutes or 600 seconds. I therefore assume that the sequence of practice from session 1 to session 16 occurred at time 0, 37.5, 75, 112.5, 150, 187.5, 225, 262.5, 300, 337.5, 375, 412.5, 450, 487.5, 525, and 562.5, respectively.

Taking the above assumptions into account, the activation of an item B'_n at the completion of its n practices is applied to compute the *model reaction time* for the $(n+1)^{th}$ practice using the ACT-R *reaction time equation* $RT'_{n+1} = I + Fe^{(-f*B'_n)}$, given $n \geq 1$. In this equation, $B'_n = ln\left(k \sum_{j=1}^{n} t_j^{-d_j}\right)$.

Effect of Proactive Interference

I now discuss a scenario on how the effect of *proactive interference* on spatial learning in Ehret's study is modelled using my model. As an example, I take the *arbitrary* label condition of Ehret's study where the buttons in the circle are labelled with icons (Figure 4.4). Each icon is arbitrarily associated with a colour.

The *mean search costs* measured by Ehret in the *arbitrary* label condition were 5.27, 2.93, 2.58, 2.34, 2.31, 1.61, 1.49, 1.31, 1.36, 1.14, 1.37, 1.15, 1.14, 1.15, and 1.08 corresponding to the sessions 1 to 15, respectively. I extracted these mean values by digitizing the graph in Figure 3a of Ehret (2002). Using the Distractor Cost equation

$X_j = E_j - 1$, for j^{th} session, $1 \leq j \leq 15$, I find the *mean Distractor Cost* (i.e. mean number of distractors) X_j at j^{th} session to be 4.27, 1.93, 1.58, 1.34, 1.31, 0.61, 0.49, 0.31, 0.36, 0.14, 0.37, 0.15, 0.14, 0.15, and 0.08 for the first 15 sessions. The mean number of distractors encountered in the first session is 4.27. This value of 4.27 affects the recall of the item in the second session. Next, the mean number of distractors encountered in the second session is 1.93. This value of 1.93 affects the recall of the item in the third session, and so on. In summary, distractors encountered in j^{th} practice session affects the recall of the item in the $(j+1)^{th}$ practice session. Finally, the mean number of distractors encountered in the 15^{th} session is 0.08, which affects the recall of the item in the 16^{th} session. I conjecture that this decreasing mean number of distractors from session to session reflects the decreasing effect of *proactive interference* on recall in the subsequent session.

Prediction of mean decay rate

The effect of *proactive interference* is evident across all three access conditions—*textual*, *arbitrary*, and *invisible*. I show an example of computing the mean *decay rate* for the first 3 sessions in the *arbitrary* label condition. Once session 1 is complete, the mean decay rate is $d_1 = h + 0.5 * X_1/N \Rightarrow d_1 = 0.058 + 0.5 * 4.27/12 \approx 0.236$. Similarly, at the end of session 2 the mean decay rate is $d_2 = h + 0.5 * X_2/N \Rightarrow d_2 = 0.058 + 0.5 * 1.93/12 \approx 0.139$ and at the end of session 3 the mean decay rate is $d_3 = h + 0.5 * X_3/N \Rightarrow d_3 = 0.058 + 0.5 * 1.58/12 \approx 0.124$. In the same way, the *decay rates* corresponding to the other practice sessions, i.e. session 4 to 15, are computed for the *arbitrary* label condition.

Using the same method, decay rates corresponding to each of the first 15 sessions are computed for *textual* and *invisible* label conditions as well.

As apparent from the decay rate equation, a change in the number of distractors changes the decay rate. While modelling the proactive interference, I noticed that the mean number of distractors per item X_j in the decay rate equation influences the model reaction time at each session-point along the abscissa. A small change in the decay rate d_j (at the level of 0.1) has impact on the reaction time predictions. This is particularly true for the first few sessions of practice—for example, second and third sessions in the present case.

Comparison of Mental Effort

First the equations $B_n' = ln\left(k \sum_{j=1}^{n} t_j^{-d_j}\right)$ and $RT_{n+1}' = I + Fe^{(-f*B_n')}$ are set up in an MS Excel spreadsheet. Next, I assume that a cut-off minimum value of R^2 is ***not*** provided for any of the three access conditions. I also assume that a tolerable maximum RMSE is ***not*** provided for any of the three access conditions. Given this constraint, I took the values of f at an increment of 0.01 in the range $0 < f \leq 1$. Furthermore, I took the values of k at an increment of 0.01 in the range $0 < k \leq 1$.

The fixed value of f is then determined following the steps in the section titled ***Procedure to determine the f value and the k values*** described earlier. Figure 4.6 shows the graph of the *sum of minimum RMSEs* of the data-fit versus *f*. For a set of chosen values of f in the range $0 < f \leq 1$, the *sum of minimum RMSEs* for an f value is obtained by adding the minimum *RMSE* of each of the three access conditions *textual*, *arbitrary* and *invisible* corresponding to that f value. Finally, the value of *f* that corresponds to the minimum value of the *sum of minimum RMSEs* of the data-fit is found to be 0.68. The $f = 0.68$ is therefore fixed across all the three access conditions.

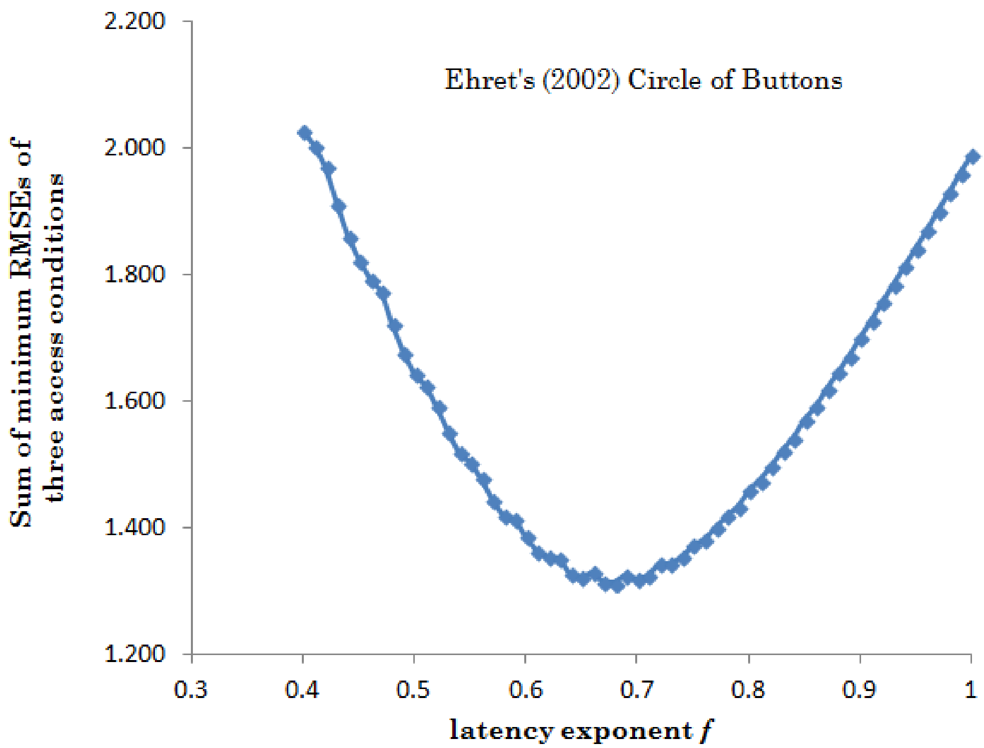

Figure 4.6 The *sum of minimum RMSEs* of the data-fit *versus* the latency exponent *f* for Ehret's (2002) study. For a set of chosen values of *f* in the range $0 < f \leq 1$, the *sum of minimum RMSEs* of the data-fit at *f* value is obtained by adding the minimum *RMSE* of each of the three access conditions *textual*, *arbitrary* and *invisible* corresponding to that *f* value. The value of $f = 0.68$ that corresponds to the minimum value of the *sum of minimum RMSEs* of the data-fit is fixed across all the three access conditions.

Once $f = 0.68$ is fixed, the k values corresponding to the three access conditions are then determined. Figure 4.7, 4.8 and 4.9 shows the graph of the RMSEs versus *effort* *factor* k for *textual*, *arbitrary* and *invisible* label conditions respectively. At each access condition, there is a value of k that corresponds to the minimum value of RMSE at $f = 0.68$. This value of k is taken to be the *effort factor* for that condition.

The k values for the three access conditions are as follows. $k = 0.74$ for the *textual* label, $k = 0.25$ for the *arbitrary* label and, $k = 0.09$ for the *invisible* label.

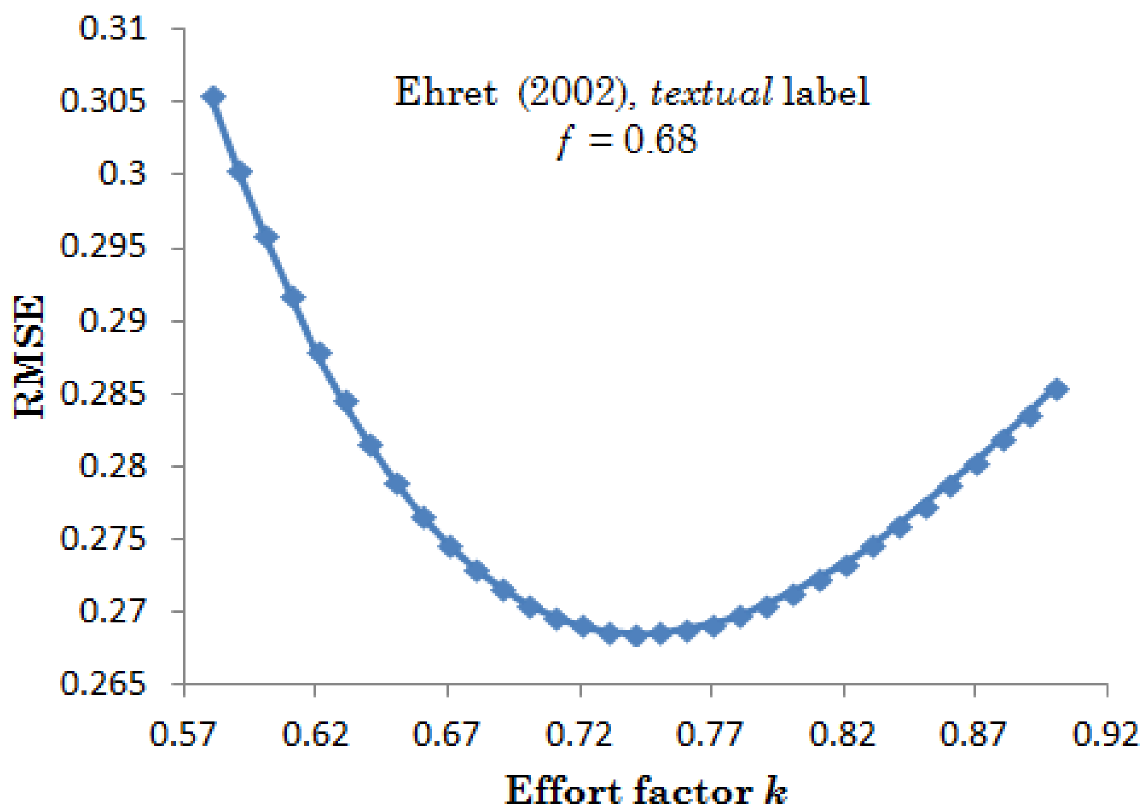

Figure 4.7 RMSE of the data-fit versus the effort factor k for the *textual* condition in Ehret's (2002) study. $k = 0.74$ corresponds to the minimum RMSE value at $f = 0.68$ for the *textual* condition. $k = 0.74$ is therefore the *effort factor* for this condition.

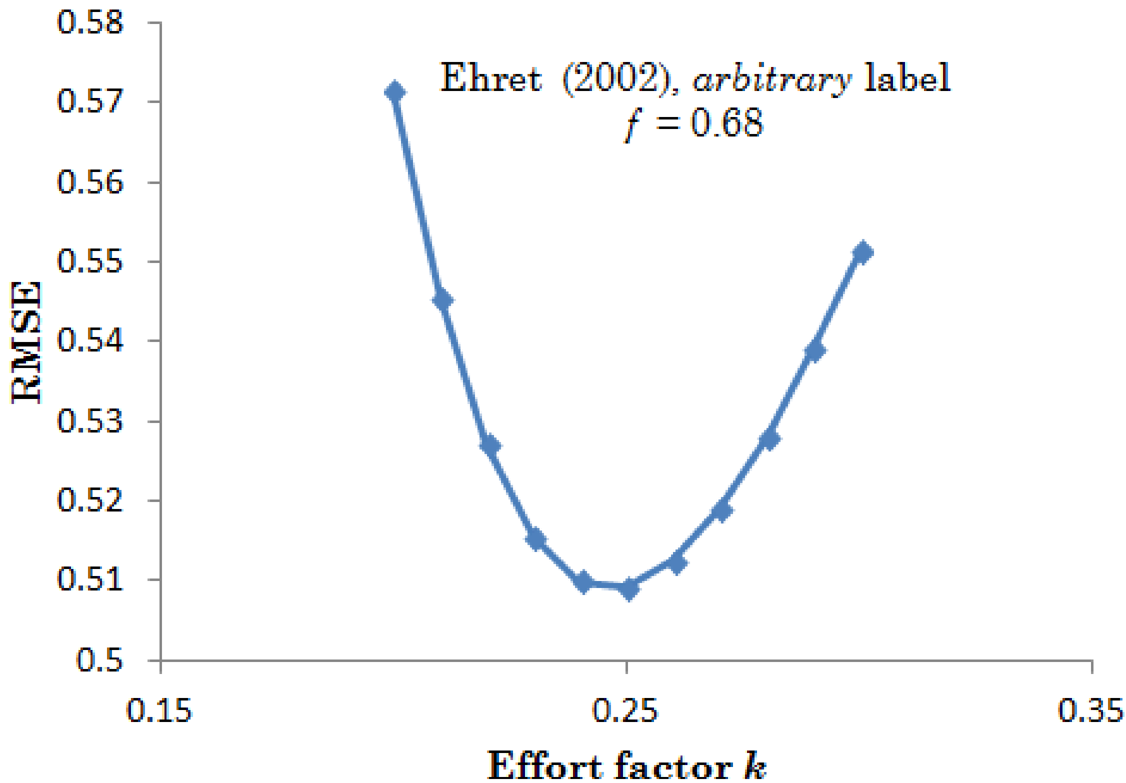

Figure 4.8 RMSE of the data-fit versus the effort factor k for the *arbitrary* condition in Ehret's (2002) study. k = 0.25 corresponds to the minimum RMSE value at f = 0.68 for the *arbitrary* condition. k = 0.25 is therefore the *effort factor* for this condition.

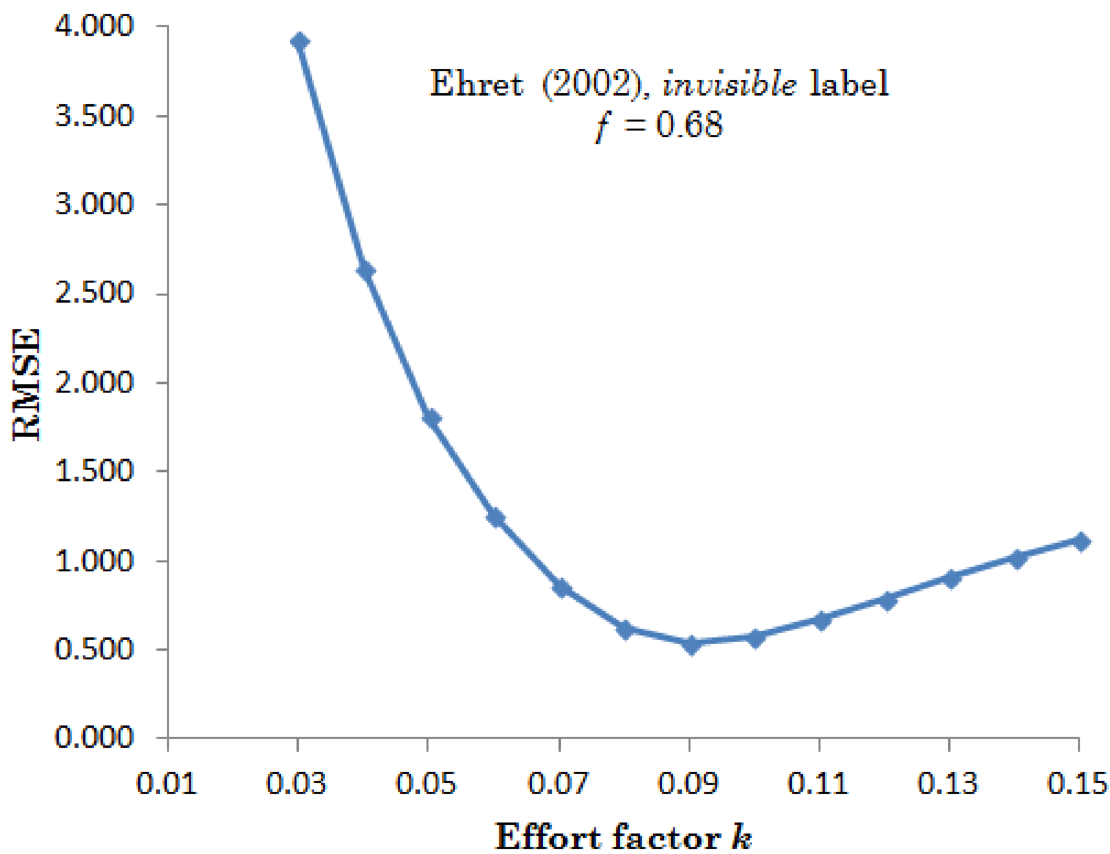

Figure 4.9 RMSE of the data-fit versus the effort factor k for the *invisible* condition in Ehret's (2002) study. k = 0.09 corresponds to the minimum RMSE value at f = 0.68 for the *invisible* condition. k = 0.09 is therefore the *effort factor* for this condition.

Figure 4.10 shows the fit of our model to the human data in terms of the *mean reaction time* to find and select a pre-cued target item (colour) in three different access conditions *textual*, *arbitrary*, and *invisible*. We compare the effort factor k for the *invisible label* condition against the *textual label* condition. We find k = 0.09 for the difficult to access *invisible* labels, compared to k = 0.74 for the easily accessible *textual* labels.

Furthermore, k is 0.25 for the difficult to access *arbitrary* labels, compared to k being 0.74 for the easy to access *textual* labels. Both instances thus point to *lower* values of *k* for access conditions of *higher* costs (i.e. lower label representativeness), compared to the access conditions where relevant information is easily available in the environment. The higher *k* value of the *arbitrary* access condition compared to that of the *invisible* access condition also suggest that the layout in *arbitrary* condition would need *less* mental effort to learn compared to the effort required to learn a layout with no labels.

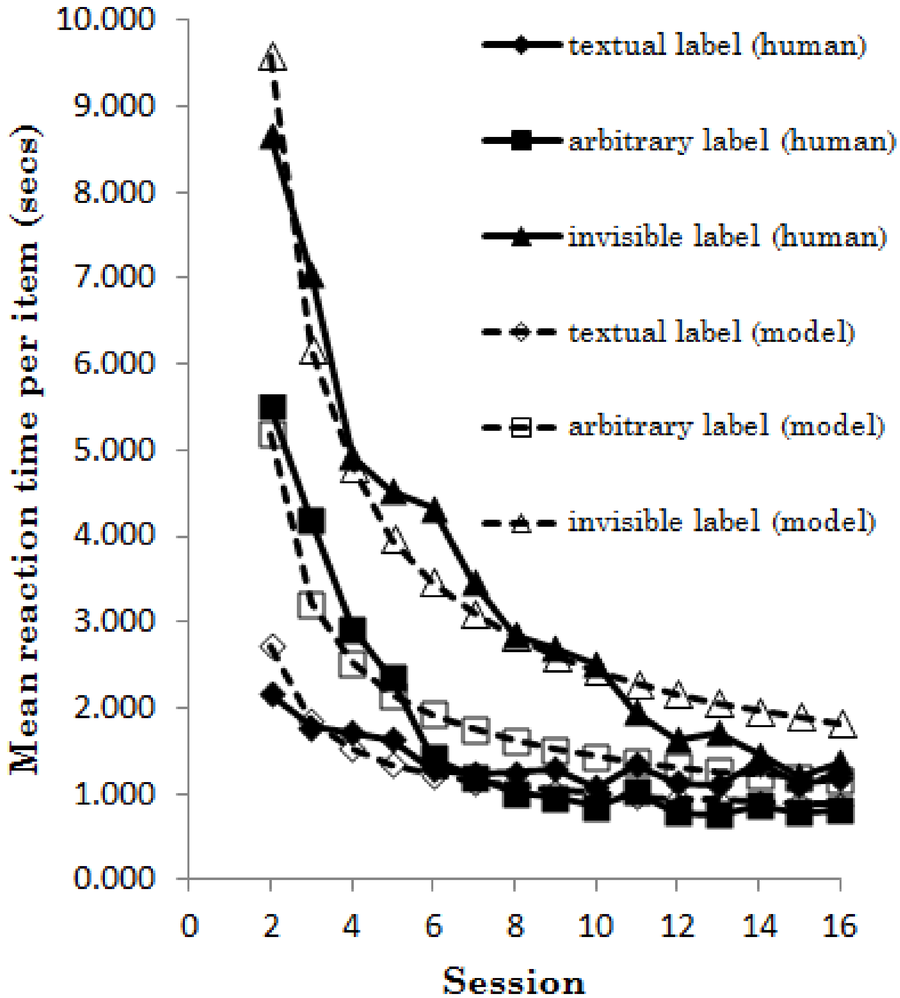

Figure 4.10 Mean reaction time per item (button) across different practice sessions for *textual*, *arbitrary* and *invisible* label conditions. Solid lines show experimental data from Ehret (2002). Dashed lines show model data predicted from the model developed in this chapter.

Table 4.1 shows the R^2, RMSE and k values for the three access conditions. With $R^2 = 0.866$, RMSE = 0.269 for the *textual*, $R^2 = 0.948$, RMSE = 0.509 for the *arbitrary* and $R^2 = 0.937$, RMSE = 0.535 for the *invisible* conditions, the correlation between the human and model data were good.

Table 4.1 R^2, RMSE and k values for the three access conditions of Ehret (2002). The latency exponent f is fixed at 0.68.

Access condition	R^2	RMSE	k
textual	0.866	0.269	0.74
arbitrary	0.948	0.509	0.25
invisible	0.937	0.535	0.09

Overall, the decreasing sequence of k values 0.74 > 0.25 > 0.09 is linked to the gradual increase in *mental effort* from the highly meaningful textual condition (k = 0.74), to the less meaningful arbitrary condition (k = 0.25), and to the least meaningful invisible condition (k = 0.09).

Given the values f = 0.68 and k = 0.25 for the *arbitrary label* condition of Ehret (2002), a sample set of *model reaction times* for item acquisition has been computed in Appendix C using the modified ACT-R reaction time equation $RT'_{n+1} = I + Fe^{(-f*B'_n)}$. The sample set consists of the predicted task completion times of the *second*, *third* and *fourth* practice sessions.

4.3.4 Graphical Keyboard (Cockburn et al., 2007b)

I now test my model against the empirical data of Cockburn et al. (2007b).

4.3.4.1 Cockburn et al.'s task

Cockburn et al. (2007b, Figure 2, p. 1574) used a *search and select* task. In a given instance of the task (i.e. in a trial), a participant was first presented with a graphical keyboard with 18 keys. There were two access conditions for the keyboard; *labelled* and *unlabelled*. In the *labelled* condition, the keyboard had every key labelled with a unique iconic symbol from the Microsoft Webdings font. In the *unlabelled* condition, the keys on the keyboard had no labels on them. Each key in the *unlabelled* condition was covered with *frost* which could be *brushed off* by waving the mouse cursor over the key to reveal its label. If left alone, the key gradually fades back to its original frosted state. The outline of every key was always visible across both the conditions (Cockburn et al. 2007b, p. 1572). In the absence of a *labelled* layout of the keyboard in Cockburn et al.'s paper, I show in Figure 4.11 how Webdings symbols would look if used as key labels. The structure of the keyboard in Figure 4.11 is a possible replica of the one in Cockburn et al. (Cockburn et al., 2007b; Figure 1, p. 1573) that I have assumed.

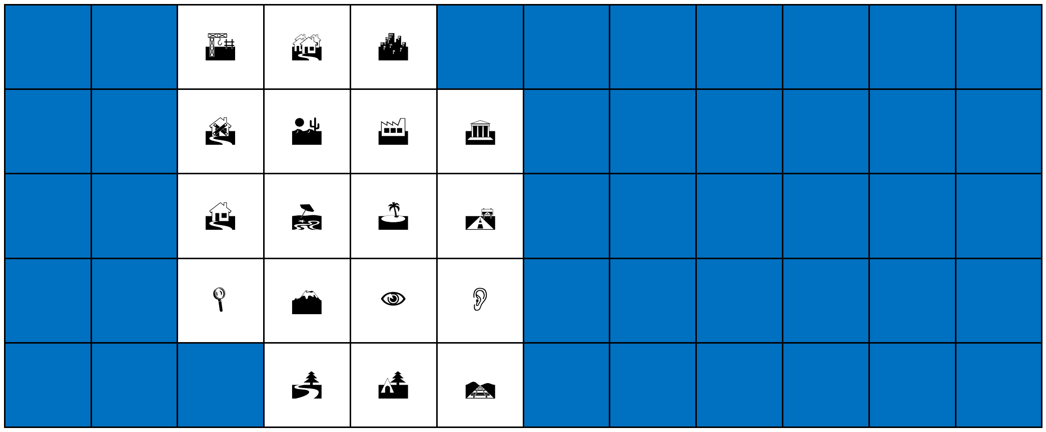

Figure 4.11 A *graphical keyboard* labelled with Webdings symbols. The structure of the keyboard is a possible replica of the one in Cockburn et al. (Cockburn et al., 2007b; Figure 1, p. 1573) that I have assumed.

All subjects practiced on both *labelled* and *unlabelled* conditions of the keyboard. Half of the subjects (Group 1) used the *unlabelled* keyboard first; the other half (Group 2) used the *labelled* keyboard first. The practice proceeded with one set of 18 symbols. The groups were then switched; and the practice proceeded again with another set of 18 symbols. Each symbol was shown in the same keyboard location for all subjects.

During the practice period, the subjects used their assigned interface (*labelled* or *unlabelled*) for 5 minutes. They were instructed that the objective was to become as efficient with the keypad as possible, and that memorising item locations would help them achieve this. There were 18 symbols displayed in a separate target-cueing region, with the next target item highlighted in green. A trial involved visually searching and selecting a pre-cued target on either layout with a computer mouse. Search for the target on keyboard was done either by *brushing* (i.e. by moving the cursor across each key under scrutiny) in case of the *unlabelled* condition or by visually searching in case of the *labelled* condition. Each successful acquisition caused a confirmation beep, and the next randomly selected symbol was highlighted green. An incorrect selection caused an error tone to be played. Subjects continued to search for the same symbol until correctly selected. *The total practice time was held constant across both conditions* (Cockburn et al., 2007b, p. 1578).

4.3.4.2 Cockburn et al.'s subjects and design

There were fourteen volunteer subjects in the study. They were all post-graduate computer science students or staff at the local university. The set of subjects included two females. In the *labelled* condition, the subjects completed the task for 10 sessions of 18 trials each. In the *unlabelled* condition, they completed the task for 5 sessions of 18 trials each.

The graphical keyboard ran in a window of fixed dimensions at 1000×600 pixels on a 15 inch 1400×1050 pixel display. The keyboard actually consisted of 60 keys (5 rows and 12 columns of keys). Out of the 60 keys, 18 were active and had a white background. The rest of the keys were inactive and were blue. A target-cuing region above the virtual keyboard showed the next target symbol, highlighted in green. It also contained a timer that showed the remaining practice time. Input was received through a high quality optical mouse. The software controlled the subject's exposure to the experimental conditions and logged all user actions.

4.3.4.3 Model Validation using human data from Cockburn et al. (2007b)

In order to validate my model I extracted two data sets from Cockburn et al.'s observations (Cockburn et al., 2007b, Figure 2, p. 1574). The two data sets correspond to the two different access conditions, *labelled* and *unlabelled*. I did this by digitizing Figure 2 of Cockburn et al.'s (2007b) work. The data sets that I derived from the digitized information are the mean *search and select* time per item (i.e. mean *task completion time* per item) for the two access conditions.

Assumptions for the data fitting exercise

For the validation of my model against the data sets, I had to make a few assumptions, as certain information was not mentioned explicitly in the work of Cockburn et al. (2007b, pp. 1573-1574). The assumptions are with respect to a given access condition, and with respect to a pre-cued target item to be found.

To find a pre-cued target item, (i) I assume that the first session occurs at time 0. (ii) I assume that the first session is equivalent to a *study practice* implying that a subject searches a layout for the target item in the first session. *No item can be recalled in the first session since the subject is scanning the layout for the very first time in that session.* (iii) I assume that recall happens from the second session onwards. Recall at a given session is affected by the number of distractors encountered in the previous session. (iv) In absence of explicit information, I assume that the acquisition of a pre-cued target item on the keyboard is accomplished by a single click of the mouse. (v) In the absence of any inter-session data (i.e. inter-trial data) in the study, I assume that the consecutive sessions were equally spaced.

Cockburn et al. (2007b, pp. 1573-1574) stated that 10 sessions took 5 minutes or 300 seconds. Therefore 5 sessions took 150 seconds. I therefore assume that the sequence of practice from session 1 to session 5 occurred at time 0, 30, 60, 90, and 120 respectively.

Taking the above assumptions into account, the activation of an item B'_n at the completion of its n practices is applied to compute the *model reaction time* for the $(n+1)^{th}$ practice using the modified ACT-R *reaction time equation* $RT'_{n+1} = I + Fe^{(-f*B'_n)}$, given $n \geq 1$. In this equation, $B'_n = ln\left(k\sum_{j=1}^{n} t_j^{-d_j}\right)$. This is the same set of equations I had used earlier to model the *task completion time* for Ehret (2002).

Effect of Proactive Interference

I now discuss a scenario on how the effect of *proactive interference* on spatial learning in Cockburn et al.'s study is modelled using my model. In my decay rate equation, the decay rates depend on the actual distractor costs across practice sessions. However, actual distractor costs across practice sessions are not provided in Cockburn et al.'s (2007b) work. For my modelling purposes, therefore, I coarsely predict the distractor costs across sessions from the *human reaction time* data reported across sessions. The objective in making this prediction is to have a reference distractor cost for each given session, so that a reference *decay rate* for each session can be predicted. This in turn helps me to reflect the relative effect of *proactive interference* across sessions.

A model to predict Distractor Cost

To obtain coarse predictions of the distractor costs across practice sessions, I assume the following: (i) Target items are not easy to discriminate from distractors. (ii) I assume *serial search* to be the visual search model for a target item. This search model proposes that attention can process only one item at a time (Horowitz & Wolfe, 1998). (iii) As per the *serial search* model, a successful search for a target will require subjects to examine, *on average*, only *half* of the items in the layout (Horowitz & Wolfe, 1998). (iv) For a given access condition, the average time spent per item during a visual search is assumed to be *constant* at every session.

Let the number of distractors encountered, i.e. the *distractor cost* at a given practice session j, $j \geq 1$ be X_j. I then predict X_j as follows:

(i) At a given practice session j, let RT_j^h is the *human reaction time* to search and select a pre-cued target item in the layout, E_j is the number of items examined during the search, and τ is the time spent per item during a search, then

$$E_j = RT_j^h/\tau \qquad \textbf{Number of Items Examined Equation}$$

As per my discussions earlier, τ is assumed to be a *constant* for a given access condition for my modelling purpose.

(ii) At the end of completion of the first practice session, the mean number of items examined E_1 is $N/2$. Here, N is the total number of items on the layout. The value $N/2$ follows from the assumption that in the *serial search* model, a successful search for a target will require subjects to examine, *on average*, only half of the items in the layout (Horowitz & Wolfe, 1998). The value of τ is therefore $\tau = (2 * RT_1^h)/N$.

(iii) The number of distractors encountered, i.e. the *distractor cost* X_j in practice session j, excludes the target item from the total number of items examined during the search. Therefore,

$$X_j = E_j - 1, \text{ where } j \geq 1 \qquad \textbf{Distractor Cost Equation}$$

This equation is an ad hoc tool to predict the *distractor cost* in a session. It just provides a rough estimate of the number of distractors at each session to differentially reflect PI across sessions in a given access condition. Next, I detail the prediction of X_j for the *labelled* and *unlabelled* conditions using my *distractor cost equation*. Note that Cockburn et al. (2007b, pp. 1573–1575) analysed only the first five practice sessions for the *unlabelled* keyboard. So, to compare the two access conditions using my model, I utilize human data only from the first five practice sessions.

Prediction of distractor cost in labelled condition

The total number of keys in the keyboard is $N = 18$. Therefore in the first session of the *labelled* condition, the mean number of items examined E_1 is $\frac{N}{2} = \frac{18}{2} = 9$. From the measured data I see that RT_1^h is 2.4 sec. Consequently the time spent per item during a search in the *labelled* condition is about $\tau = (2 * RT_1^h)/N = (2 * 2.4)/18 = 0.267$ sec. This τ value is assumed to be a constant across all the practice sessions in the *labelled* condition.

Subsequently, using the *human reaction times* RT_j^h for sessions j = 2, 3 and 4, I predict the number of keys examined E_j at those sessions. I use the *number of items examined equation* $E_j = RT_j^h/\tau$ for this purpose. For example, the number of keys examined at the second session is $E_2 = RT_2^h/\tau = 2.031/0.267 \approx 7.62$. Thus, in the first four sessions, the mean number of keys examined E_j are determined as 9.00, 7.62, 7.10 and 6.41. Then I use the *distractor cost equation* $X_j = E_j - 1$ for j = 1, 2, 3 and 4. Consequently, the mean number of distractors X_j encountered in the first four sessions is 8, 6.62, 6.10, and 5.41.

The mean number of distractors encountered in the first session is 8. This value of 8 affects the recall of the item in the second session. Next, the mean number of distractors encountered in the second session is 6.62. This value of 6.62 affects the recall of the item in the third session, and so on. In summary, distractors encountered in j^{th} practice session affects the recall of the item in the $(j+1)^{th}$ practice session. Finally, the mean number of distractors encountered in the 4^{th} session is 5.41, which affects the recall of the item in the 5^{th} session. I conjecture that this decreasing mean number of distractors from session to session reflects the decreasing effect of *proactive interference* on recall in the subsequent session.

Prediction of distractor cost in unlabelled condition

The total number of keys in the keyboard is N = 18. Therefore in the first session of the *unlabelled* condition, the mean number of items examined E_1 is $\frac{N}{2} = \frac{18}{2} = 9$. From the measured data I see that RT_1^h = 4.599 sec. Consequently the time spent per item during a search in the *unlabelled* condition is about

$\tau = (2 * RT_1^h)/N = (2 * 4.599)/18 = 0.511$ sec. This τ value is assumed to be a constant across all the practice sessions in the *unlabelled* condition. Subsequently, using the *human reaction times* RT_j^h for the sessions j = 2, 3 and 4, I predict the number of keys E_j examined at those sessions. I use the *number of items examined equation* $E_j = RT_j^h/\tau$ for this purpose. For example, the number of keys examined in the second session is $E_2 = RT_2^h/\tau = 3.171/0.511 \approx 6.21$. Thus, in the first four sessions, the mean number of keys examined E_j are determined as 9.00, 6.21, 5.39 and 4.67. Then I again use the *distractor cost equation* $X_j = E_j - 1$ for j = 1, 2, 3 and 4. Consequently, the mean number of distractors X_j encountered in the first four sessions is 8, 5.21, 4.39 and 3.67.

Similar to the scenario of *labelled* condition, the decreasing number of distractors from session to session reflects the decreasing effect of *proactive interference* on recall in the subsequent session in this *unlabelled* condition as well.

Prediction of mean decay rate

The effect of *proactive interference* is evident across the two access conditions—*labelled* and *unlabelled*. I show an example of computing the mean *decay rate* for the first 3 sessions in the *labelled* condition. Once session 1 is complete, the mean decay rate is $d_1 = h + 0.5 * X_1/N \Rightarrow d_1 = 0.058 + 0.5 * 8/18 \approx 0.280$. Similarly, at the end of session 2 the mean decay rate is $d_2 = h + 0.5 * X_2/N \Rightarrow d_2 = 0.058 + 0.5 * 6.62/18 \approx 0.242$ and at the end of session 3, $d_3 = h + 0.5 * X_3/N \Rightarrow d_3 = 0.058 + 0.5 * 6.10/18 \approx 0.227$. In the same way, the *decay rate* corresponding to session 4 is computed for the *labelled* condition.

Using the same method, decay rates corresponding to each of the first 4 sessions are computed for *unlabelled* condition.

Comparison of Mental Effort

First the equations $B'_n = ln\left(k \sum_{j=1}^{n} t_j^{-d_j}\right)$ and $RT'_{n+1} = I + Fe^{(-f * B'_n)}$ are set up in an MS Excel spreadsheet. Next, I assume that a cut-off minimum value of R^2 is ***not*** provided for any of the two access conditions. I also assume that a tolerable maximum RMSE is ***not*** provided for any of the two access conditions. Given this constraint, I took the values of f at an increment of 0.01 in the range $0 < f \leq 1$. Furthermore, I took the values of k at an increment of 0.01 in the range $0 < k \leq 1$.

The fixed value of f is then determined following the steps in the section titled ***Procedure to determine the f value and the k values*** described earlier. Figure 4.12 shows the graph of the *sum of minimum RMSEs* of the data-fit versus *f*. For a set of chosen values of f in the range $0 < f \leq 1$, the *sum of minimum RMSEs* for an f value is obtained by adding the minimum *RMSE* of each of the two access conditions *labelled* and *unlabelled* corresponding to that f value. Finally, the value of f that corresponds to the minimum value of the *sum of minimum RMSEs* of the data-fit is found to be 0.26. The f = 0.26 is therefore fixed across the two access conditions.

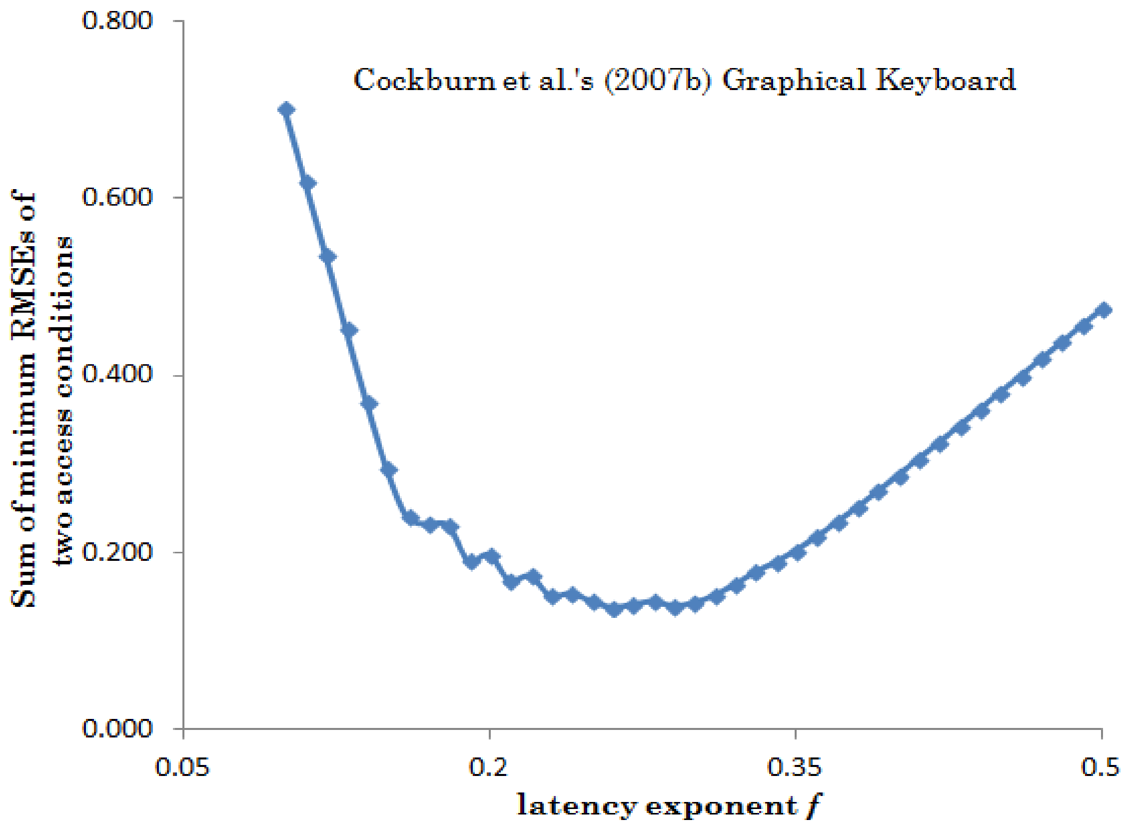

Figure 4.12 The *sum of minimum RMSEs* of the data-fit *versus* the latency exponent *f* for Cockburn et al.'s (2007b, pp. 1573-1575) graphical keyboard study. For a set of chosen values of *f* in the range $0 < f \leq 1$, the *sum of minimum RMSEs* of the data-fit at *f* value is obtained by adding the minimum *RMSE* of each of the two access conditions *labelled* and *unlabelled* corresponding to that *f* value. The value of *f* = 0.26 that corresponds to the minimum value of the *sum of minimum RMSEs* of the data-fit is fixed across the two access conditions.

Once $f = 0.26$ is fixed, the k values corresponding to the two access conditions are then determined. Figure 4.13 and 4.14 shows the graph of the RMSEs versus *effort* *factor k* for the *labelled* and *unlabelled* conditions of the keyboard respectively. For each access condition, there is a value of k that corresponds to the minimum value of RMSE at $f = 0.26$. This value of k is taken to be the *effort factor* for that condition.

The k values for the two access conditions are as follows. $k = 0.39$ for the *labelled* keyboard, and $k = 0.06$ for *unlabelled* keyboard.

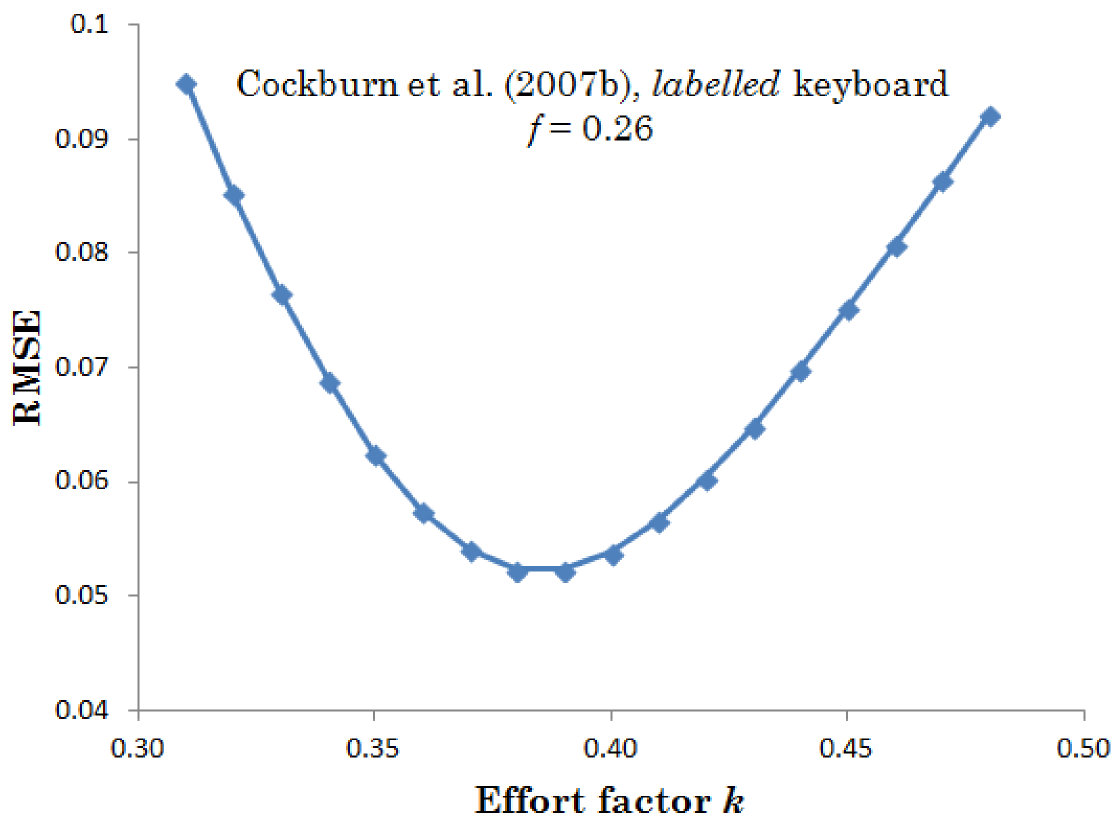

Figure 4.13 RMSE of the data-fit versus the effort factor *k* for the *labelled* condition in Cockburn et al.'s (2007b, pp. 1573-1575) graphical keyboard study. *k* = 0.39 corresponds to the minimum RMSE value at *f* = 0.26 for the *labelled* condition. k = 0.39 is therefore the *effort factor* for this condition.

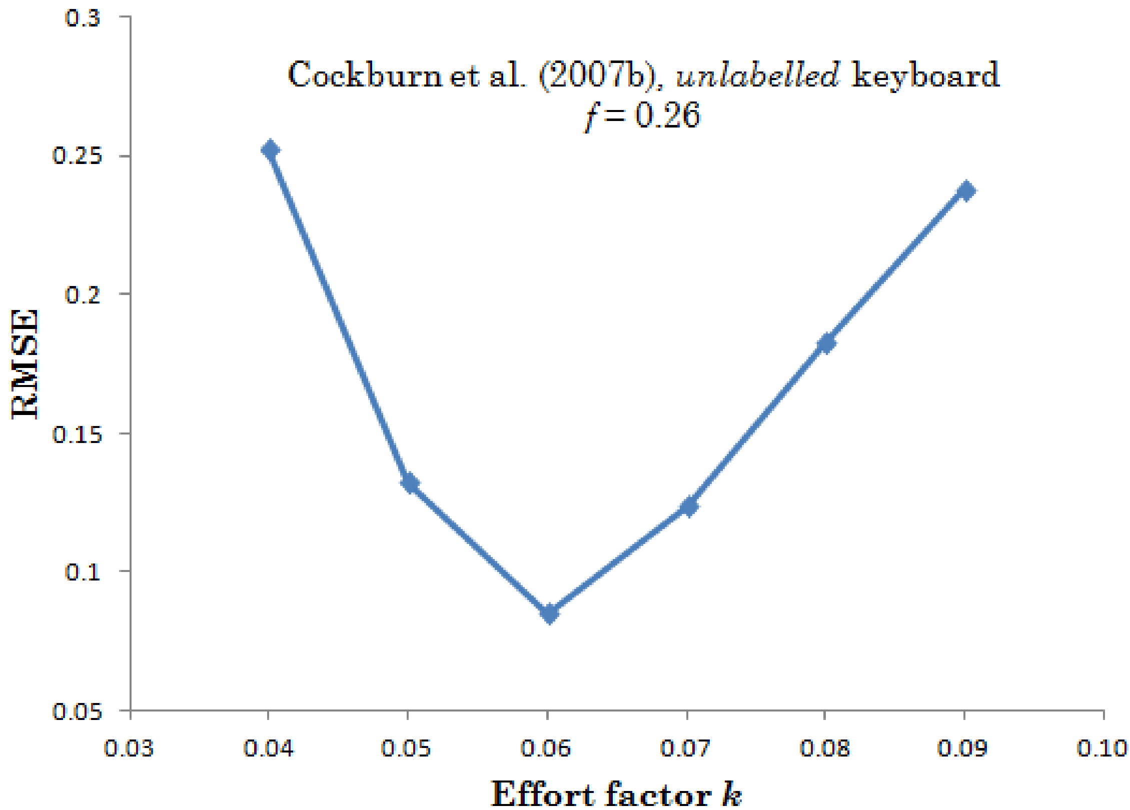

Figure 4.14 RMSE of the data-fit versus the effort factor *k* for the *unlabelled* condition in Cockburn et al.'s (2007b, pp. 1573-1575) graphical keyboard study. *k* = 0.06 corresponds to the minimum RMSE value at *f* = 0.26 for the *unlabelled* condition. *k* = 0.06 is therefore the *effort factor* for this condition.

Figure 4.15 shows the fit of our model to the human data in terms of the *mean reaction time* to find and select a pre-cued target item (symbol) in two different access conditions *labelled* and *unlabelled*. We compare the effort factor *k* for the *unlabelled* condition against the *labelled* condition. We find *k* = 0.06 for the difficult to access *unlabelled* keyboard, compared to *k* = 0.39 for the easily accessible *labelled* keyboard.

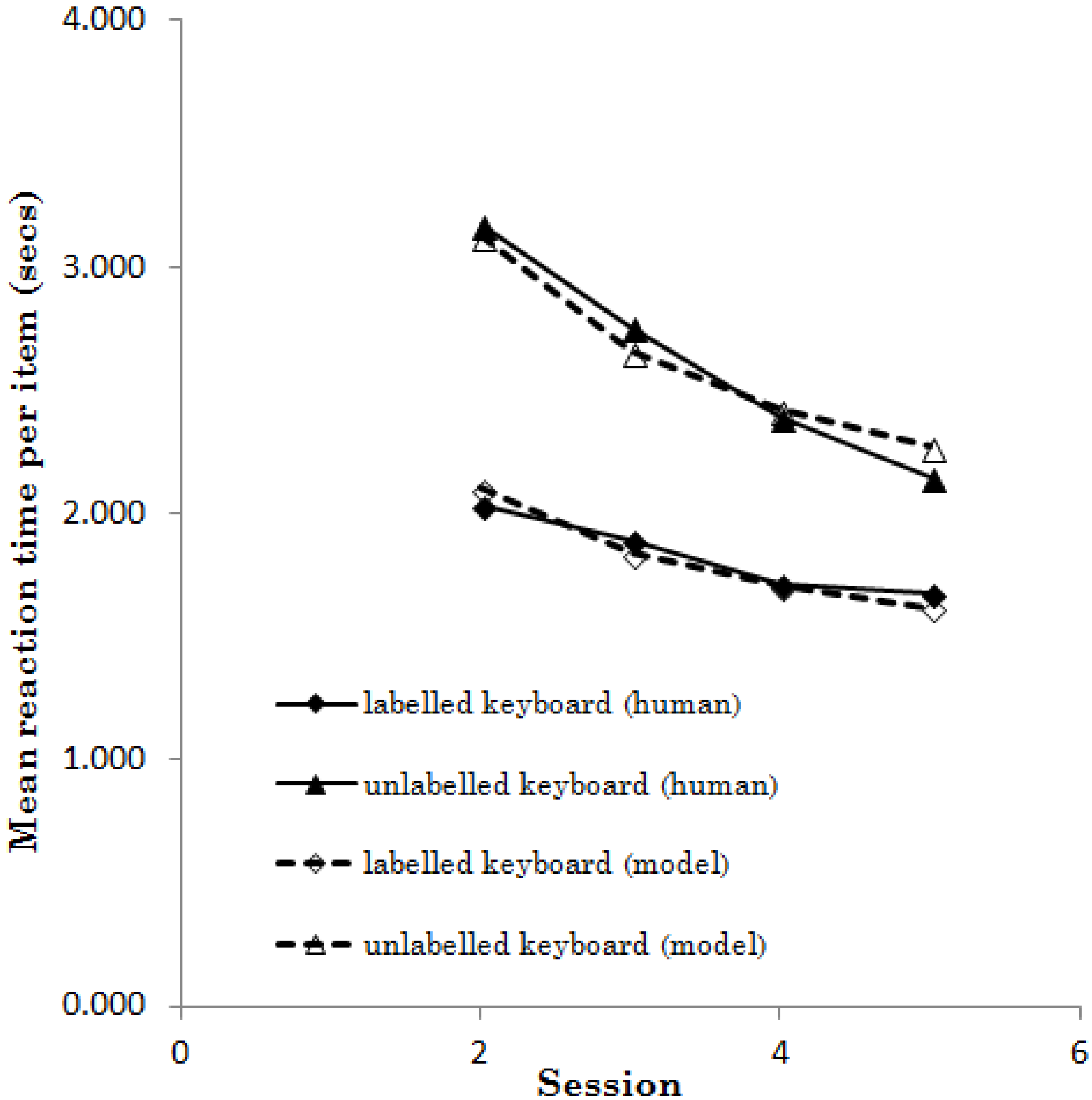

Figure 4.15 Mean reaction time per item (symbol) across different practice sessions for the *labelled* and *unlabelled* conditions of the graphical keyboard. Solid lines show experimental data from Cockburn et al. (2007b, pp. 1573-1575). Dashed lines show model data predicted from the model developed in this chapter.

The *lower* value of *k* reflects an access condition of *higher* cost (higher perceptual cost) compared to a condition, where relevant information is easily available in the environment. The higher *k* value of the *labelled* condition compared to that of the *unlabelled* condition also suggest that a layout with labels would need *less mental* *effort* to learn compared to the one with no labels. This is similar to my model's validation against Ehret's (2002) human data performed earlier.

Table 4.2 shows the R^2, RMSE and k values for the two access conditions. With R^2 = 0.955, RMSE = 0.052 for the *labelled* and R^2 = 0.975, RMSE = 0.085 for the *unlabelled* conditions, the correlation between the human and model data were good.

Table 4.2 R^2, RMSE and k values for the two access conditions in Cockburn et al.'s (2007b, pp. 1573-1575) graphical keyboard study. The latency exponent f is fixed at 0.26.

Access condition	R^2	RMSE	k
labelled	0.955	0.052	0.39
unlabelled	0.975	0.085	0.06

4.4 Discussion and Conclusions

In this chapter, I proposed a closed-form model of spatial learning that is able to quantitatively distinguish between different levels of effortful conditions due to different label representativeness of layouts. The model combines the effect of practice in terms of *age of practice*, the effect of mental effort in terms of an *effort factor*, the effect of proactive interference in terms of *distractor cost* (i.e. number of distractors), and the effect of decay in terms of a *numeric constant*—all together into a single equation of memory activation.

Similar to my first model, this model can also predict the future reaction times of a given layout. The prediction of the reaction time at a given session is possible provided that a mechanism to obtain the number of distractors of prior sessions is available.

I extended the existing base-level activation equation of ACT-R theory for my purpose. I validated my model against previous empirical data sets. Others collected these data sets by observing the process of learning stable graphical layouts whose item configurations were initially unfamiliar to the participants. The tasks involved searching and selecting pre-cued items on the layouts using a mouse. I found good agreement of my model with the empirically gathered data for comparing access conditions that differed from each other.

My work in this chapter introduces two mathematical constructs. One is the *decay rate equation* to account for the effect of *proactive interference* and the other is the *effort factor* to account for the effect of *mental effort*. I include them in an existing memory activation equation of ACT-R theory that hitherto accounted only for the effects of *practice* and *decay*.

While comparing a given set of layouts in terms of their *mental effort*, all the free model parameters h, f and F are kept the *same* except the *effort factor* k. The effort factor k was the *only* free parameter that varies to reflect the differences in the *mental effort* across different effortful conditions (i.e. access conditions) in the given set.

As part of my model validation process, I separately compared two *sets* of layouts. The first set consisted of three circular layouts that differed in their label representativeness (Ehret, 2002). The second set consisted of two rectangular keyboard layouts that also differed in their label representativeness (Cockburn et al., 2007b). For each of the two data sets, I used What-If analysis of MS Excel to find the f value per data set and the k values. My model predictions matched the human data with high R^2 values (greater than 0.85 for the first data set and greater than 0.95 for the second data set), and low RMSE values (less than 0.55 for the first data set and less than 0.09 for the second data set)—see Table 4.1 and 4.2.

I used my model to compare different layouts that contain the same number of items. However my model is general enough to compare different types of layouts containing different number of items.

My closed-form model based on ACT-R declarative memory equations has its limitations. (i) Unlike a simulation model, it is unable to express the progression of interaction between cognitive modules over time. (ii) Unlike a simulation model, it is unable to account for the noise in the activation levels. (iii) At any given trial for searching a target location on a layout, if the number of distractors X_j encountered is much less than the total number of items N on the layout, I assume that proactive interference in that trial has been negligible. This situation may arise when N is very large. Further investigation is warranted to identify a practical upper limit on N. (iv) My model does not account for the effect of *visual similarity* between the distractors and the target on proactive interference. (v) ACT-R theory has a threshold parameter that specifies a minimum activation below which an item is not retrievable by the cognitive system. Similar to Altmann & Schunn (2002), I assume no such threshold. As the threshold parameter is not a variable in the equations I use, this assumption does not impact my work directly.

Overall, my closed-form model saves substantial expertise, labour and time that may have been spent in developing a low-level description of a simulation model (e.g. Freed & Remington, 2000; Paik, Kim, Ritter et al., 2010). Yet, it enables me to obtain a coarse but quick prediction of the relative differences in *mental effort* required to learn different layouts.

Kim and Ritter (in press) suggest that a *high effort* condition promotes *short-term retention* whereas a *low effort* condition promotes *long-term retention*. They also suggest that a *low effort* condition promotes *quick relearning*. Since the effort factor *k* of my second model can quantitatively distinguish between a high and a low effort interface, it can help identify interfaces that would promote short-term retention, long-term retention or quick relearning.

Chapter 5 Conclusions

The *goal* of this thesis was to develop simulative and closed-form cognitive models of learning interactive layouts. I developed two models.

The first model is a simulation model of text copying on a traditional phone keypad. It leverages a mathematical equation to model visual exploration instead of implementing a low-level simulation custom search module. The mathematical equation expresses the transition from *search* to *choice*. The transition is governed by the level of *recall accuracy* of a learner. The second model is a closed-form model that synthesizes the effect of *practice*, *memory decay*, *proactive interference*, and *mental effort* on task completion time.

5.1 My first model

My first model is a simulation model. It predicts the learning of a traditional phone keypad layout (Figure 3.3) through a text copying task. I choose a text copying task because text copying is a skill that requires a great deal of learning or training (Cockburn et al., 2007b; p. 1571). Such a task substantially consumes one's cognitive and perceptual time, especially in the early stages of learning (see for example, Salthouse, 1986; John, 1996; Kim, Ritter & Koubek, 2013; Kim & Ritter, in press).

I tested the novice part of my model's prediction against human data. The human data contained considerable oscillations. Therefore I tested the difference between the slopes of the regression lines of novice human and model data. For $\alpha = 0.05$ (two-tailed), I concluded I have no reason to doubt that the mean non-Fitts time decreases as a function of practice sessions at the same rate for human as for model.

My first model could be useful to predict a learning curve of any given layout constrained with a maximum number of items. A learning curve prediction can provide answers to several important questions related to the design of layouts. For example, it can provide insight into how fast item acquisition can be at a given stage of learning, which stage a learner is in, and how much practice is needed by a learner to reach the expert level. These answers may save valuable training time and cost and help to allocate resources effectively.

Considering letter frequency in learning curve prediction for text entry

Currently my model executes the task of copying a group of 5 distinct English letters in a given session. These 5 letters are randomly chosen out of 26 letters. Here, it is assumed that the frequency for each of the 26 letters is the same. In reality, the occurrence frequencies for letters are different in human languages. To accommodate this difference in letter frequencies in English, the driver that controls the simulated experiment needs to change. The change should be such that the letters are chosen depending on their frequency. As a consequence, higher frequency letters will be chosen more often than the lower frequency letters in a given session. This will impact the *Non Exploration Time* (NET) and *Recall Accuracy* (RA) per session. Consequently the predicted learning curve will be different than the same frequency case.

In the ensuing discussion, I assume that *only* one label is mapped to one interactive item. This is to simplify my explanation.

<u>*Predicting a learning curve of another layout using the first model*</u>

My model could be generalized to predict a learning curve of an item acquisition task on other layouts. Such layouts can be different from the traditional phone keypad. For example, the layout can be a graphical layout. In this regard, the following information should be provided: the *practice schedule*; the *number of items* on a layout; the human *Visual Search Time* (VST), that is, the human *Visual Exploration Time* (VET) for the *first session* to find an item; and the human non-Fitts time (NFT) for first few sessions. Moreover, the *layout configuration* itself is necessary to develop the simulative sub-model.

First, the simulative sub-model based on classic ACT-R should be constructed to account for visual encoding and memory retrieval. This sub-model will predict NET and RA for each session. Second, the *choice reaction time* (CRT) time should be predicted by substituting the number of items in Hick's Law assuming that the Hick's Law constants are known. Third, the NFT for each session is to be predicted using the equation NFT = NET + VET, where VET = (1 – *RA*) * VST + *RA* * CRT. Note that an equation similar to this VET equation has been used earlier by Cockburn et al. (2007a) to predict the *visual exploration time* for different linear menu layouts, and by Ahlstrom et al. (2010) to predict the *visual exploration time* for spatially stable layouts such as matrix menu layouts, pie menu layouts and traditional linear menu layouts. Their equation, however, is not based on any cognitive principles (Cockburn & Gutwin, 2010, p. 13:5). In contrast, the recall accuracy *RA* is one term in my VET equation that accounts for cognition. Fourth, the two ACT-R parameters, the *retrieval threshold* and the *latency factor* are tuned to match the human data of a first few sessions. Subsequently, the future reaction times can be predicted. My model needs to be tested in this regard.

<u>Predicting a learning curve of a hierarchical layout using the first model</u>

My first model can be generalized to predict the performance of item acquisition in a hierarchical layout. For each layout in the hierarchy, a single level learning curve can be predicted following the approach described earlier in the section titled *Predicting a learning curve of another layout using the first model.*

Finally, the total *non-Fitts time* spent to find a target at a given level of hierarchy, in a given session, can be predicted by summing two components at that session: the non-Fitts time at that level and the non-Fitts times of all prior levels. This idea of summing for hierarchical layouts is guided by the work of Ahlstrom et al. (2010, p. 1374). Investigation is recommended in this regard to test model predictions.

<u>Limitations of my first model</u>

Here, I list the limitations of my first model.

I tested the novice part of my model's prediction against the human data. However, the *progression* along the learning curve from novice to expert level is yet to be validated.

My model does not account for the effect of potential errors that may be committed by entering unexpected characters while copying text. Modification of the current model to accommodate the effect of such errors is not a straightforward task. Future investigation is warranted in this regard.

In my model, the *minimum* VET is represented by the *choice reaction time* (CRT) for a button in the phone keypad. This is to model the VET of an expert user. In a key-pressing task on a keyboard, Seibel (1963) observed that the *choice reaction time* increased for 2 to approximately 8 alternatives, and showed trivial further increase no matter how many additional alternatives were added to the task. Thus, being dependent on the *choice reaction time*, my model becomes constrained by the limitation of a maximum of 8 alternative items.

Although the aforementioned restriction related to the *choice reaction time* may be a disadvantage for modelling and analyzing location learning on large screens, such as laptops where more than 8 items are not uncommon, it may be appropriate for analysis of small-screen layouts, such as those found in cell phones and PDAs. Besides, in recent years Cockburn and associates (Cockburn et al., 2007a; Ahlstrom et al., 2010; Cockburn & Gutwin 2010) observed and modelled the *choice reaction time* for interaction with up to 12 alternative graphical buttons using mouse on computer screens. Similar to my first model, they used Hick's Law for modelling the *choice reaction time*. Thus, I speculate that if my model is generalized for graphical layouts, it will get constrained by the limitation of a maximum of 12 alternative graphical buttons.

My simulation sub-model is limited in that it does not incorporate the repeated key presses required to arrive at a letter on a traditional phone keypad (see Figure 3.3

for the layout). For example, to copy the character sequence `cei`, the user needs to press the key containing `c` only once instead of pressing it thrice, and so on. I do this to stay compatible with the specific user study of Pavlovych and Stuerzlinger (2004) that I validated my model against.

My first model focuses purely on the cognitive aspects of interaction; it does not model the motor control complexities involved in spatial search and selection processes on user interfaces. In reality though, these are all important factors that influence the overall user experience.

5.2 My second model

My second model is a closed-form model. It accounts for the combined effect of *practice*, *mental effort*, *proactive interference* and *decay* along the three stages of learning. The primary reason to develop this model is to *quantitatively* compare multiple interactive layouts in terms of the *mental effort* required to learn them. The layouts differ in their label representativeness.

I validated my model against previous empirical data sets of learning graphical layouts. The tasks involved searching and selecting pre-cued items on the layouts using a mouse.

My model predictions matched human data with high R^2 values (greater than 0.85 for the first data set and greater than 0.95 for the second data set), and low RMSE values (less than 0.55 for the first data set and less than 0.09 for the second data set).

For the same layout, the effort factor *k* may differ depending on the set of layouts being compared. This is due to the restriction on the latency exponent *f* that is constant for the set. *f* is selected to minimize the sum of RMSEs across all layouts.

I used my model to compare different layouts that contain the same number of items. However my model is general enough to compare different types of layouts containing different numbers of items.

Being a closed form model, my second model has some advantages—it simplifies the model description in comparison to a simulation model that is normally specified in the unwieldy low-level notation of a contemporary cognitive architecture (e.g. Paik, Kim, Ritter et al., 2010). It is computationally inexpensive, less complex and more straightforward to apply than a simulation model. Developing an analogous simulation model could require substantial time and expertise (Freed & Remington, 2000).

My second model can differentiate between a low effort interface versus a high effort one. The *effort factor k* for a layout provides insight into the relative *mental effort* expended to learn—the higher the value of *k*, the lower is the *mental effort*; the lower the value of *k*, the higher is the *mental effort*. Such comparison may be applicable in different situations. Some examples are as follows.

Effortfulness, Retention and Relearning

The Soft Constraint Hypothesis postulates that a *high effort* learning condition (knowledge in-the-head) promotes memory-intensive strategies (Gray et al., 2006).

Kim and Ritter (in press) suggest that a memory-intensive strategy may be forgotten more with longer retention intervals unless the knowledge is proceduralized. In contrast, a *low effort* learning condition (knowledge in-the-world) promotes interaction-intensive strategies (Gray et al., 2006). Kim and Ritter (in press) suggest that an interaction-intensive strategy promotes *long-term retention* as well as *quick relearning*.

Kim and Ritter (in press) suggest that the above conclusions are applicable in choosing user interfaces in areas that involve learning by human operators. For example, the learning of surgical task knowledge by medical students. The students progress through a learning curve to reach expertise. During this progression, they might forget some task knowledge they had learned and they might want to conserve memory. As a result, they may resort to an interaction-intensive strategy instead of a memory-intensive strategy. On the other hand, if they interact often and interaction time is important, supporting a memory-intensive strategy or both strategies simultaneously would become important.

Since the effort factor k of my second model can quantitatively distinguish between a high and a low effort interface, it can help choose interfaces that would be useful in the scenario explained above.

Effort and Ego Depletion

Baumeister, Bratslavsky, Muraven, & Tice (1998) suggest that an effortful condition may lead to *ego depletion*. Ego depletion refers to the *depletion of self-control*. Self-control draws from a limited resource. When one consumes energy from this limited resource, subsequent acts of self-control become impaired (Baumeister et al., 1998). In a recent work, Subramanium (2011) observed that execution of a *difficult* task led to a reduced ability to forgo immediate small rewards for delayed larger rewards compared to the execution of an *easy* task.

The aforementioned suggestion and observation allow me to speculate that learning an interface in a *higher effort condition* (difficult condition) may result in *higher ego depletion*, which in turn may result in higher performance degradation in a *subsequent* difficult task. In contrast, learning an interface in a *lower effort condition* (easy condition) may result in *lower ego depletion*, which in turn may result in lower performance degradation in a *subsequent* difficult task.

Given the learning curves of multiple interfaces for a given task, my second model can be used to rank the effort required to accomplish the task on each interface. This ranking can be done in terms of the *effort factor* values. Thus, effort factor values could inform which interface would deplete *more* self-control and which one would deplete *less*.

Factors affecting effortful conditions

In my second model, I considered label representativeness as a factor for different effortful conditions. My model can be generalized to distinguish the effortfulness caused by different input modalities (e.g. Kim & Ritter, in press).

System delay is another factor that may cause different effortful conditions (Golightly et al., 1999). Further investigation is necessary to generalize my second model in this regard.

Different spacings of practice events may also cause different effortful conditions (Pavlik & Anderson, 2005, as cited in Pavlik 2007). Differentiating such effortful conditions may be necessary to answer questions such as for a given layout, which spacing—mass, distributed, or a combination of both—would require higher effort to learn. To reflect the effortfulness due to differences in spacing, further investigation of my second model is recommended.

Effect of visual similarity on proactive interference

Underwood (1957) suggests that the lower the *visual similarity*[9] between the distractors and the target item, the lower is the *proactive interference*. My second model does not account for the effect of *visual similarity* between the distractors and the target on proactive interference (PI). One way to account for the effect of *visual similarity* on PI could be through an optional mechanism in ACT-R declarative memory known as *partial matching*.

In the *partial matching* mechanism, the values of attributes of an item to be recalled are attempted for a "close enough" match with the values of corresponding attributes of items in the memory—for example, the *colour* attribute value orange could be considered to be somewhat close to the *colour* attribute value red. The modeller can

[9] Here, I mean *similarity* with respect to the basic attributes (Wolfe & Horowitz, 2004; p. 6), such as colour, shape, size and orientation.

also specify a numerical penalty for such matches, on a case-by-case basis—the closer the match, the higher is the penalty. This penalty value is subtracted from the base-level activation of the item. In ACT-R, these penalties must be manually specified (Stewart et al., 2007, p. 231). These penalties can be used to reflect the PI due to similarities—the closer the match, the higher is the penalty, and the lower is the activation of the item in question. Therefore the *difficulty* to recall the item is higher, reflecting a higher PI. To leverage the benefit of the *partial matching* mechanism, my modified base-level activation equation needs to be merged in the ACT-R simulation framework. Further investigation is recommended in this regard.

Limitations of my second model

A limitation of my second model arises due to the fact that it is not a simulation model. It is unable to express the progression of interaction between the cognitive modules over time. Further, it is unable to account for a recall failure. Unlike a simulation model, it is also unable to account for noise in the activation levels.

Another limitation of my second model is as follows: At any given trial for searching a target location on a layout, if the number of distractors X_j encountered is much less than the total number of items N on the layout, I assume that proactive interference in that trial has been negligible. This situation may arise when N is very large. Further investigation is warranted to identify a practical upper limit on N.

Chapter 6 References

Agresti, A., & Finlay, B. (1997). *Statistical Methods For the Social Sciences* (3rd ed.). Upper Saddle River, NJ: Prentice Hall.

Ahlstrom, D., Cockburn, A., Gutwin, C., & Irani, P. (2010). 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* (pp. 1371–1380).

Altmann, E. M., & Gray, W. D. (2008). An integrated model of cognitive control in task switching. *Psychological Review, 115*(3), 602–639.

Altmann, E. M., & Schunn, C. D. (2002). Integrating decay and interference: A new look at an old interaction. In *Proceedings of the 24th annual meeting of the Cognitive Science Society* (pp. 65–70).

Anderson, J. R. (1974). Retrieval of propositional information from long-term memory. *Cognitive Psychology*, *6*(4), 451–474.

Anderson, J. R. (1983). A spreading activation theory of memory. *Journal of Verbal Learning and Verbal Behavior*, *22*(3), 261–295.

Anderson, J. R. (1991). Is human cognition adaptive? *Behavioral and Brain Sciences*, *14*(03), 471–485.

Anderson, J. R., Bothell, D., Byrne, M. D., Douglass, S., Lebiere, C., & Qin, Y. (2004). An integrated theory of the mind. *Psychological Review*, *111*(4), 1036–1060.

Baumeister, R. F., Bratslavsky, E., Muraven, M., & Tice, D. M. (1998). Ego depletion: is the active self a limited resource? *Journal of Personality and Social Psychology*, *74*(5), 1252–1265.

Bogacz, R., Usher, M., Zhang, J., & McClelland, J. L. (2007). Extending a biologically inspired model of choice: multi-alternatives, nonlinearity and value-based multidimensional choice. *Philosophical Transactions of the Royal Society B*, *362*, 1655–1670.

Busemeyer, J. R., & Diederich, A. (2010). *Cognitive modeling, Chapter 1*. Sage.

Butts, L., & Cockburn, A. (2002). An evaluation of mobile phone text input methods. *Australian Computer Science Communications*, *24*(4), 55–59.

Byrne, M. D. (2001). ACT-R/PM and menu selection: Applying a cognitive architecture to HCI. *International Journal of Human-Computer Studies*, *55*(1), 41–84.

Card, S. K., English, W. K., & Burr, B. J. (1978). Evaluation of mouse, rate-controlled isometric joystick, step keys, and text keys for text selection on a CRT. *Ergonomics*, *21*(8), 601–613.

Card, S. K., Moran, T. P., & Newell, A. (1986). *The psychology of human-computer interaction*. CRC Press.

Casner, S., & Larkin, J. H. (1989). Cognitive efficiency considerations for good graphic design. In *Proceedings of the 11th Annual Conference of the Cognitive Science Society* (pp. 275–282).

Cochran, R. E., Lee, F. J., & Chown, E. (2006). Modeling emotion: Arousal's impact on memory. In *Proceedings of the 28th Annual Conference of the Cognitive Science Society* (pp. 1133–1138).

Cockburn, A., & Gutwin, C. (2010). A model of novice and expert navigation performance in constrained-input interfaces. *ACM Transactions on Computer-Human Interaction (TOCHI)*, *17*(3), 13:1–13:38.

Cockburn, A., Gutwin, C., & Greenberg, S. (2007a). A predictive model of menu performance. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (pp. 627–636).

Cockburn, A., Kristensson, P. O., Alexander, J., & Zhai, S. (2007b). Hard lessons: effort-inducing interfaces benefit spatial learning. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (pp. 1571–1580).

Cockburn, A., & Siresena, A. (2003). Evaluating mobile text entry with the Fastap keypad. In *Proceedings of the 17th British HCI Group Annual Conference* (pp. 77–80).

Das, A., & Stuerzlinger, W. (2007). A cognitive simulation model for novice text entry on cell phone keypads. In *Proceedings of the 14th European conference on Cognitive Ergonomics* (pp. 141–147).

Das, A., & Stuerzlinger, W. (2008). Modeling learning effects in mobile texting. In *Proceedings of the 7th international Conference on Mobile and Ubiquitous Multimedia* (pp. 154–161).

Das, A., & Stuerzlinger, W. (2010). Proactive interference in location learning: A new closed-form approximation. In *Proceedings of the 10th international conference on cognitive modeling* (pp. 37–42).

Das, A., & Stuerzlinger, W. (2012). Comparing cognitive effort in spatial learning of text entry keyboards and ShapeWriters. In *Proceedings of the International Working Conference on Advanced Visual Interfaces* (pp. 649–652).

Das, A., & Stuerzlinger, W. (2013). Unified Modeling of Proactive Interference and Memorization Effort: A new mathematical perspective within ACT-R theory. In

Proceedings of the 35th Annual Conference of the Cognitive Science Society (pp. 358–363).

Dunlop, M. D., & Crossan, A. (1999). Dictionary based text entry method for mobile phones. In *Second Workshop on Human-Computer Interaction with Mobile Devices* (pp. 5–7).

Dunlop, M. D., & Crossan, A. (2000). Predictive text entry methods for mobile phones. *Personal Technologies*, *4*(2-3), 134–143.

Ehret, B. D. (1999). *Learning where to look: The acquisition of location knowledge in display-based interaction* (PhD thesis). George Mason University, Fairfax, Virginia, USA.

Ehret, B. D. (2002). Learning where to look: Location learning in graphical user interfaces. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (pp. 211–218).

Elmes, D. G. (1988). Interference in spatial memory. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, *14*(4), 668–675.

Freed, M., & Remington, R. (2000). Making human-machine system simulation a practical engineering tool: An APEX overview. In *Proceedings of the International Conference on Cognitive Modeling* (pp. 110–117).

Golightly, D., Hone, K. S., & Ritter, F. E. (1999). Speech interaction can support problem solving. In *Proceedings of IFIP TC.13 International Conference on Human–Computer Interaction* (pp. 149–155).

Grant, D. A. (1962). Testing the null hypothesis and the strategy and tactics of investigating theoretical models. *Psychological Review*, *69*(1), 54–61.

Gray, W. D., & Fu, W.-T. (2001). Ignoring perfect knowledge in-the-world for imperfect knowledge in-the-head. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (pp. 112–119).

Gray, W. D., & Fu, W.-T. (2004). 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.

Gray, W. D., Sims, C. R., Fu, W.-T., & Schoelles, M. J. (2006). The soft constraints hypothesis: a rational analysis approach to resource allocation for interactive behavior. *Psychological Review*, *113*(3), 461–482.

Halverson, T., Gunzelmann, G., Moore Jr, L. R., & Van Dongen, H. P. (2010). Modeling the effects of work shift on learning in a mental orientation and rotation task. In *Proceedings of the International Conference on Cognitive Modeling. Philadelphia, PA: Drexel University* (pp. 79–84).

Hick, W. E. (1952). On the rate of gain of information. *Quarterly Journal of Experimental Psychology*, *4*(1), 11–26.

Horowitz, T. S., & Wolfe, J. M. (1998). Visual search has no memory. *Nature*, *394*(6693), 575–577.

Howell, D. C. (2007). *Statistical methods for psychology* (6th ed.). Belmont, CA: Thomson Wadsworth.

Ivory, M. Y., & Hearst, M. A. (2001). The State of the Art in Automating Usability Evaluation of User Interfaces. *ACM Comput.Surv.*, *33*(4), 470–516.

James, C. L., & Reischel, K. M. (2001). Text input for mobile devices: comparing model prediction to actual performance. In *Proceedings of the SIGCHI Conference on*

Human Factors in Computing Systems (pp. 365–371).

John, B. E. (1996). TYPIST: A theory of performance in skilled typing. *Human-Computer Interaction*, *11*(4), 321–355.

John, B. E., Prevas, K., Salvucci, D. D., & Koedinger, K. (2004). Predictive human performance modeling made easy. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (pp. 455–462).

Just, M. A., & Carpenter, P. A. (1992). A capacity theory of comprehension: Individual differences in working memory. *Psychological Review*, *99*, 122–149.

Keppel, G., & Underwood, B. J. (1962). Proactive inhibition in short-term retention of single items. *Journal of Verbal Learning and Verbal Behavior*, *1*(3), 153–161.

Kim, J. W., & Ritter, F. E. (in press). Learning, Forgetting, and Relearning for Keystroke-and Mouse-Driven Tasks: Relearning is Important. *Human-Computer Interaction*.

Kim, J. W., Ritter, F. E., & Koubek, R. J. (2013). An integrated theory for improved skill acquisition and retention in the three stages of learning. *Theoretical Issues in Ergonomics Science*, *14*(1), 22–37.

Kotovsky, K., & Simon, H. A. (1990). What makes some problems really hard: Explorations in the problem space of difficulty. *Cognitive Psychology*, *22*(2), 143–183.

Kveraga, K., Boucher, L., & Hughes, H. C. (2002). Saccades operate in violation of Hick's law. *Experimental Brain Research*, *146*, 307–314.

Lawrence, B. M., St. John, A., Abrams, R. A., & Snyder, L. H. (2008). An anti-Hick's effect in monkey and human saccade reaction times. *Journal of Vision*, *8*(3), 1–7.

MacKenzie, I. S. (1992). Fitts' law as a research and design tool in human-computer interaction. *Human-Computer Interaction*, *7*(1), 91–139.

MacKenzie, I. S., & Buxton, W. (1992). Extending Fitts' law to two-dimensional tasks. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (pp. 219–226).

MacKenzie, I. S., & Soukoreff, R. W. (2002). Text entry for mobile computing: Models and methods, theory and practice. *Human-Computer Interaction*, *17*(2-3), 147–198.

Mayers, A. (2013). *Introduction to Statistics and SPSS in Psychology* (1st ed.). Canada: Pearson Education.

McKenna, S., & Glendon, A. (1985). Occupational first aid training: Decay in cardiopulmonary resuscitation (CPR) skills. *Journal of Occupational Psychology*, *58*(2), 109–117.

Newell, A. (1990). *Unified theories of cognition* (Vol. 1987). Cambridge, Mass.: Harvard University Press.

Norman, D. A. (1988). *The design of everyday things*. Basic books.

O'Hara, K. P., & Payne, S. J. (1998). The effects of operator implementation cost on planfulness of problem solving and learning. *Cognitive Psychology*, *35*(1), 34–70.

Paik, J., Kim, J. W., Ritter, F. E., Morgan, J. H., Haynes, S. R., & Cohen, M. A. (2010). Building large learning models with Herbal. In *Proceedings of the International Conference on Cognitive Modeling* (pp. 187–192).

Pavlik Jr, P. I. (2007). Timing is an order: Modeling order effects in the learning of information. *In Order to Learn: How the Sequences of Topics Affect Learning*, 137–150.

Pavlik Jr., P. I., Presson, N., & Koedinger, K. R. (2007). Optimizing knowledge component learning using a dynamic structural model of practice. In *Proceedings of the International Conference on Cognitive Modeling* (pp. 37–42).

Pavlik, P. I., & Anderson, J. R. (2005). Practice and Forgetting Effects on Vocabulary Memory: An Activation-Based Model of the Spacing Effect. *Cognitive Science*, *29*(4), 559–586.

Pavlik, P. I., & Anderson, J. R. (2008). Using a model to compute the optimal schedule of practice. *Journal of Experimental Psychology: Applied*, *14*(2), 101–117.

Pavlovych, A., & Stuerzlinger, W. (2003). Less-Tap: A Fast and Easy-to-learn Text Input Technique for Phones. In *Proceedings of Graphics Interface* (pp. 97–104).

Pavlovych, A., & Stuerzlinger, W. (2004). Model for non-expert text entry speed on 12-button phone keypads. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (pp. 351–358).

Peterson, L., & Peterson, M. J. (1959). Short-term retention of individual verbal items. *Journal of Experimental Psychology*, *58*(3), 193–198.

Pew, R. W., & Mavor, A. S. (2007). *Human-system integration in the system development process: A new look*. National Academies Press.

Raaijmakers, J. G. W. (2003). Spacing and repetition effects in human memory: Application of the SAM model. *Cognitive Science*, *27*(3), 431–452.

Ritter, F. E., Baxter, G., Kim, J. W., & Srinivasmurthy, S. (2013). Learning and retention. In *John D. Lee and Alex Kirlik (eds.). The Oxford Handbook of Cognitive Engineering* (pp. 125–142). New York, NY: Oxford.

Ritter, F. E., & Kim, J. W. (2010). ACT-R: Frequently Asked Questions List. Retrieved from http://acs.ist.psu.edu/act-r-faq/act-r-faq4-10.html

Ritter, F. E., & Larkin, J. H. (1994). Developing process models as summaries of HCI action sequences. *Human-Computer Interaction*, *9*(4), 345–383.

Ritter, F. E., Schoelles, M. J., Quigley, K. S., & Klein, L. C. (2011). Determining the number of simulation runs: Treating simulations as theories by not sampling their behavior (pp. 97–116). Springer.

Rowe, G., Hasher, L., & Turcotte, J. (2008). Age differences in visuospatial working memory. *Psychology and Aging*, *23*(1), 79–84.

Salthouse, T. A. (1986). Perceptual, cognitive, and motoric aspects of transcription typing. *Psychological Bulletin*, *99*(3), 303–319.

Sears, A., Jacko, J. A., Chu, J., & Moro, F. (2001). The role of visual search in the design of effective soft keyboards. *Behaviour & Information Technology*, *20*(3), 159–166.

Seibel, R. (1963). Discrimination reaction time for a 1,023-alternative task. *Journal of Experimental Psychology*, *66*(3), 215–226.

Seow, S. C. (2005). Information theoretic models of HCI: a comparison of the Hick-Hyman law and Fitts' law. *Human-Computer Interaction*, *20*(3), 315–352.

Silfverberg, M., MacKenzie, I. S., & Korhonen, P. (2000). Predicting text entry speed on mobile phones. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (pp. 9–16).

Soukoreff, R. W., & Mackenzie, I. S. (1995). Theoretical upper and lower bounds on typing speed using a stylus and a soft keyboard. *Behaviour & Information*

Technology, *14*(6), 370–379.

Soukoreff, R. W., & MacKenzie, I. S. (2004). Towards a standard for pointing device evaluation, perspectives on 27 years of Fitts' law research in HCI. *International Journal of Human-Computer Studies*, *61*(6), 751–789.

St. Amant, R., Horton, T. E., & Ritter, F. E. (2007). Model-based evaluation of expert cell phone menu interaction. *ACM Transactions on Computer-Human Interaction (TOCHI)*, *14*(1), 1–24.

Stewart, T. C., & West, R. L. (2007). Deconstructing and reconstructing ACT-R: Exploring the architectural space. *Cognitive Systems Research*, *8*(3), 227–236.

Subramaniam, S. (2011). *Ego depletion and delay of gratification* (M.Sc. Thesis). Georgia Southern University, Statesboro, Georgia, USA.

Taatgen, N., & Rijn, H. V. (2010). Nice graphs, good R2, but still a poor fit? How to be more sure your model explains your data. In *Proceedings of the International Conference on Cognitive Modeling* (pp. 247–252).

Treisman, A. M., & Gelade, G. (1980). A feature-integration theory of attention. *Cognitive Psychology*, *12*(1), 97–136.

Underwood, B. J. (1957). Interference and forgetting. *Psychological Review*, *64*(1), 49.

Welford, A. T. (1968). Fundamentals of skill.

West, R. L., Pyke, A. A., Rutledge-Taylor, M., & Lang, H. (2010). Interference and ACT-R: New evidence from the fan effect. In *Proceedings of the International Conference on Cognitive Modeling* (pp. 211–216).

Wickens, C. D., & Hollands, J. G. (2000). *Engineering psychology and human performance* (3rd ed.). Upper Saddle River, NJ: Prentice Hall.

Wolfe, J. M., & Horowitz, T. S. (2004). What attributes guide the deployment of visual attention and how do they do it? *Nature Reviews Neuroscience*, *5*(6), 1–7.

Appendix A
Disclaimer

The following peer-reviewed materials are related to the work presented in this dissertation.

Das, A., & Stuerzlinger, W. (2007). A Cognitive Simulation Model for Novice Text Entry on Cell Phone Keypads. In *Proceedings of the European Conference on Cognitive Ergonomics: ECCE 2007* (pp. 141-147). London, UK. (**Chapter 3**)

Das, A., & Stuerzlinger, W. (2008). Modeling Learning Effects in Mobile Texting. In *Proceedings of the 7th International Conference on Mobile and Ubiquitous Multimedia: MUM 2008* (pp. 154-161). Umea, Sweden. (**Chapter 3**)

Das, A. & Stuerzlinger, W. (2010). Proactive interference in location learning: A new closed-form approximation. In *Proceedings of the 10th International Conference on Cognitive Modeling: ICCM 2010* (pp. 37-42). Philadelphia, PA. (**Chapter 4**)

Das, A., & Stuerzlinger, W. (2012). Comparing cognitive effort in spatial learning of text entry keyboards and ShapeWriters. *Poster session presented* at *the International Working Conference on Advanced Visual Interfaces: AVI 2012* (pp. 649-652). Capri Island, Italy. (**Chapter 4**)

Das, A., & Stuerzlinger, W. (2013). Unified Modeling of Proactive Interference and Memorization Effort: A new mathematical perspective within ACT-R theory. In *Proceedings of the 35th Annual Conference of the Cognitive Science Society: CogSci 2013* (pp. 358-363). Austin, TX. (**Chapter 4**)

Appendix B
The rules of the simulation model of Chapter 3

I assume that there are three main areas on the frontal surface of the cell phone handset: *display area*, *text output area* and *keypad area* from top to bottom respectively. See Figure 3.2. Text displayed in the *display area* is copied to the *text output area* by pressing relevant keys present in the *keypad area*.

The simulation model of Chapter 3 uses a single modeller-defined ACT-R chunk-type. The chunks created from the chunk-type help the model to keep track of (i) its state; (ii) the location of the current letter on the display area; (iii) the last letter keyed in; (iv) the current letter to be keyed in; and (v) the location of the current letter on the keypad.

I represent the procedural knowledge of the simulation model of Chapter 3 using the following nineteen production rules:

- *seek-location-of-first-char-on-phrase* finds the position of the first letter of the displayed phrase in the field of view.
- *switch-attention-on-phrase* shifts the visual attention to the position of the current letter that has been found on the displayed phrase.
- *extract-stimulus-on-phrase* encodes the attended letter on the displayed phrase so that the letter becomes accessible to the model.

- *char-does-not-equal-last-char-on-phrase* matches if the current letter found is not same as the last letter. If a match occurs, it attempts to retrieve the keypad position of the current letter from the declarative memory. If the retrieval is successful, the retrieval buffer (associated with declarative memory) is filled up with the chunk containing the letter and its coordinates on keypad. If the retrieval is not successful, the retrieval buffer becomes empty. This rule is tried only for the retrieval of the keypad position of the first letter of each letter-group.
- *recall-location-on-keypad* matches if the keypad coordinates of the current letter (that has just been encoded from the displayed phrase) is same as the information present in the retrieval buffer and fails to match if it doesn't. If the match occurs, the model will execute a motor action directly, without any attention shift, to enter the letter.
- *cannot-recall-location-on-keypad* matches if the keypad coordinates of the current letter (that has just been encoded from the displayed phrase) is not same as the information present in the retrieval buffer (more specifically when the retrieval buffer is empty). If the match occurs, it will lead to the shift of visual attention, to the keypad area, for the current letter.
- *char-equals-last-char-on-phrase* matches when the current letter found is same as the last letter. If the match occurs, a motor action is carried out.
- *seek-location-of-char-on-keypad* finds the position of the current letter (that has just been encoded from the displayed phrase) on the keypad when the letter's position on the keypad cannot be recalled from declarative memory.
- *switch-attention-on-keypad* shifts the visual attention to the position of the current letter that has been found on the keypad.
- *extract-stimulus-on-keypad* encodes the attended letter on keypad so that the letter becomes accessible to the model.

- *do-peck-for-first-char-of-the-phrase* executes a peck movement for the first letter of the phrase.
- *prepare-for-thumb-recoil* gets the model ready for recoiling the right thumb.
- *do-thumb-recoil-before-peck-for-next-char* enables the model to recoil its right thumb to the recoil home location (3, 2) shown in Figure 3.3.
- *do-peck-for-next-char-of-the-phrase-after-thumb-recoil* enables the model to execute a peck movement for the relevant letter of the phrase (except the first letter of the phrase). In our case, this rule will apply to the first letter of every letter-group (except for the first group).
- *do-punch-when-char-equals-last-char* enables the model to execute a punch movement when the letter to be entered is same as the last letter entered.
- *get-location-of-current-char-on-phrase* retrieves the location of the current letter in the displayed phrase from declarative memory. Note that this production only helps in getting the thread of control back to the phrase from the keypad after each letter is entered. This production/transfer between foci of attention cannot be avoided and adds an overhead of 50 ms for every letter entered.
- *seek-location-of-next-char-on-phrase* attempts to find the position of the next letter of the phrase.
- *end-of-phrase-not-reached-yet* is fired when the position of a new letter in the phrase has been found. In that case, the visual attention is shifted to the position of the newly found letter. At this point, the execution continues.
- *end-of-phrase-reached* is fired when there are no more letters left to be read in the phrase. At this point, the execution stops.

Appendix C
A sample computation of the predicted task completion time related to Chapter 4

This appendix is related to the closed-form model of Chapter 4. Here, I show a sample computation of the predicted task completion time for the second, third and fourth practice sessions in case of the *arbitrary* label condition in *circle of buttons* (Ehret, 2002).

Total number of buttons on the circle N = 12. The model parameters are fixed as follows. F = 1, h = 0.058, f = 0.68, k = 0.25, I = 0.595 (that is, visual encoding time for the item + movement time to the location of the item + time to click on the item = 0.085 sec + 0.360 sec + 0.150 sec = 0.595 sec).

Ehret (2000, p. 136) had expressed that 16 sessions took 10 minutes or 600 seconds. I therefore assume that the sequence of practice from session 1 to session 4 occurred at time 0, 37.5, 75 and 112.5.

The first practice occurs at session 1. The mean number of distractors encountered is $X_1 = 4.27$. Due to the first practice, the mean decay rate is $d_1 = h + 0.5 * X_1/N \Rightarrow d_1 = 0.058 + 0.5 * 4.27/12 \approx 0.236$. The second practice is to occur at session 2. The second practice is to be impacted by the base-level activation of the item just before the second practice starts. The base-level activation of the item just before the second practice starts is computed using the modified base-level activation equation

$B_1' = ln\left(k\sum_{j=1}^{1} t_j^{-d_j}\right)$ as follows. $\sum_{j=1}^{1} t_j^{-d_j} = 37.5^{-0.236}$. Therefore, $B_1' = ln\left(k\sum_{j=1}^{1} t_j^{-d_j}\right)$ = $ln($ 0.25 $* 37.5^{-0.236})$ = -2.241. Next, I use the modified ACT-R reaction time equation, $RT_{n+1}' = I + F * e^{(-f*B_n')}$ to predict the task completion time at the end of second practice as follows. $RT_2' = I + F * e^{(-f*B_1')} = 0.595 + 1 * e^{(-0.68 * -2.241)}$ = 5.186 sec.

The second practice occurs at session 2. The mean number of distractors encountered is $X_2 = 1.93$. Due to second practice, the mean decay rate is $d_2 = h + 0.5 * X_2/N \Rightarrow d_2 = 0.058 + 0.5 * 1.93/12 \approx 0.139$. The third practice is to occur at session 3. The third practice is to be impacted by the base-level activation of the item just before the third practice starts. The base-level activation of the item just before the third practice starts is computed using the modified base-level activation equation $B_2' = ln\left(k\sum_{j=1}^{2} t_j^{-d_j}\right)$ as follows. $\sum_{j=1}^{2} t_j^{-d_j} = 75^{-0.236} + 37.5^{-0.139}$. This is because just before the third practice starts at session 3, the age of the first practice is 75 and the age of second practice is 37.5. Therefore, $B_2' = ln\left(k\sum_{j=1}^{2} t_j^{-d_j}\right) = ln($ 0.25 * $[75^{-0.236} + 37.5^{-0.139}])$ = -1.420. Next, I predict the task completion time at the end of third practice as follows. $RT_3' = I + F * e^{(-f*B_2')} = 0.595 + 1 * e^{(-0.68 * -1.420)}$ = 3.222 sec.

The third practice occurs at session 3. The mean number of distractors encountered is $X_3 = 1.58$. Due to third practice, the mean decay rate is $d_3 = h + 0.5 * X_3/N \Rightarrow d_3 = 0.058 + 0.5 * 1.58/12 \approx 0.124$. The fourth practice is to occur at session 4. The fourth practice is to be impacted by the base-level activation of the item just before the fourth practice starts. The base-level activation of the item just before the fourth practice starts is computed using the modified base-level activation equation $B_3' = ln\left(k\sum_{j=1}^{3} t_j^{-d_j}\right)$ as follows. $\sum_{j=1}^{3} t_j^{-d_j} = 112.5^{-0.236} + 75^{-0.139} + 37.5^{-0.124}$. This is because just before the fourth practice starts at session 4, the age of the first practice is 112.5, the age of second practice is 75 and the age of third practice is 37.5. Therefore, $B_3' = ln\left(k\sum_{j=1}^{3} t_j^{-d_j}\right) = ln(\ 0.25 * [\ 112.5^{-0.236} + 75^{-0.139} + 37.5^{-0.124}\ ])$ = -0.970. Next, I predict the task completion time at the end of fourth practice as follows. $RT_4' = I + F * e^{(-f * B_3')} = 0.595 + 1 * e^{(-0.68 * -0.970)}$ = 2.529 sec.