

Päivi Majaranta

Text Entry by Eye Gaze

ACADEMIC DISSERTATION

To be presented with the permission of the Faculty of Information Sciences of the
University of Tampere, for public discussion in Pinni auditorium B1097
on August 1st, 2009, at noon.

Department of Computer Sciences
University of Tampere

Dissertations in Interactive Technology, Number 11
Tampere 2009

ACADEMIC DISSERTATION IN INTERACTIVE TECHNOLOGY

Supervisor: Professor Kari-Jouko Rähkä, Ph.D.,
Department of Computer Sciences,
University of Tampere,
Finland

Opponent: Prof. Dr. phil. habil. Anke Huckauf,
Juniorprofessor of Psychophysiology and Perception,
Bauhaus-University of Weimar,
Germany

Reviewers: Associate Professor Anthony Hornof, Ph.D.,
Department of Computer and Information Science,
University of Oregon,
USA

Assistant Professor Hiroataka Aoki, Ph.D.,
Tokyo Institute of Technology,
Japan

Dissertations in Interactive Technology, Number 11

Department of Computer Sciences
FIN-33014 University of Tampere
FINLAND

ISBN 978-951-44-7786-7
ISSN 1795-9489

Tampereen yliopistopaino Oy
Tampere 2009

Abstract

Text entry by eye gaze is used by people with severe motor disabilities. An eye tracking device follows the user's eye movements, and a computer program analyzes the gaze behavior. To type by gaze, the user typically points at the characters on an on-screen keyboard by looking at them and selects them by means of dwell time, a prolonged gaze that separates an intentional command from casual viewing.

The basic methods for producing text by gaze have been researched and in real-world use since the early 1980s; however, the design issues have not been studied in detail. Until recently, assistive eye tracking systems were used mostly by a small number of people who were totally paralyzed and for whom gaze control was a necessity and the only option. The technology and its usability have improved considerably, and several new systems have appeared on the market, making the technology available for a much wider group of users with varying need and abilities. Today, the eye tracker can be considered an optional assistive device worth considering since it provides easy and fast access to information technology by gaze alone.

This thesis provides an extensive review of the research conducted in the area of gaze-based text entry. It summarizes results from several experiments that study various aspects of text entry by gaze. Results show that small improvements in the interface design can lead to significant improvements in user performance and satisfaction. For example, adding a simple "click" that confirms the selection by gaze can significantly improve the text entry speed over that of plain visual feedback. The improvement is small, but the effect accumulates in the repetitive task of text entry.

An overview of different design solutions and guidelines derived from the research results are given. It is hoped that the thesis will provide a useful starting point for developers, researchers, and assistive technology professionals wishing to gain deeper insight into gaze-based text entry.

Acknowledgements

I find it hard to believe that I actually did it. And the truth is that it is not I but *we* who did it. All of the work reported in this thesis has been done in close collaboration with my colleagues and co-authors. Your efforts are highly appreciated!

First and foremost I want to thank Kari-Jouko Räihä, my supervisor, co-author, and a trusted friend. Kari is the leader of the TAUCHI research unit where I work and the head of the UCIT graduate school that has partly supported this thesis work. He is also the coordinator of the IST Network of Excellence on Communication by Gaze Interaction (COGAIN), funded by the European Commission, where I have had the pleasure of working for the last five years. Without COGAIN, I would never have gained the level of insight into gaze interaction that I now have.

I want to express special thanks to my co-author, Scott MacKenzie, who taught me how to do experimental research and how to revise overlong manuscripts into half of the original length without losing any of the actual content.

In addition to Kari and Scott, I owe a great deal to all of my other co-authors, each of whom had an essential role in this work. Anne Aula, Oleg Špakov, Outi Tuisku, Ulla-Kaija Ahola, Niina Majaranta, Richard Bates, Mick Donegan, Gintautas Daunys, and Poika Isokoski – thank you!

There are two colleagues who truly deserve a big thank you, Stina Boedeker and Aulikki Hyrskykari, with whom I have had the pleasure of working closely for many years. You have listened to my gripes and groans and given me the daily support when I have needed it.

I would also like to thank all of my colleagues and the administrative staff at the Department of Computer Sciences for providing facilities for the work. Especially, Tuula Moisio, thank you for teaching me how to navigate in the jungle of travel bills and other similar stuff that I cannot even name here. Jori Mäntysalo, thank you for providing technical support 24/7 for the COGAIN web server and for patiently explaining Linux jargon over and over again. Members of the VIRG research group, thank you for motivating discussions! Thanks to the working mates who found the time to attend the always relaxing, sometimes even inspiring coffee break discussions.

I also wish to thank the reviewers of the thesis, Anthony Hornof and Hirotaka Aoki, for their constructive comments.

Finally, the most important thanks go to my family. Leo, Niina, Jenni, thank you for your patience over these long years! You are the joy of my life! Leena, Olavi, Eija, Pauli, Teija, thank you for your constant support!

Tampere, June 29th, 2009
Päivi Majaranta

Articles

This thesis is based on several research articles published in journals, conference proceedings, and book chapters. Some portions are reproduced as they are, while others have been revised and updated for the present publication. Text passages from some articles have also been rearranged to better suit the structure of this thesis, and overlapping portions have been removed or shortened. Explicit permission has been acquired from the publishers as well as from the co-authors.

I am the main and first author of most papers. However, all papers contain essential contributions from the co-authors. A brief overview of the chapters of this thesis is given below, with full references to the original articles used as the basis for the chapters, accompanied with a brief note on my personal contribution to each paper.

The first piece contributing to this thesis was a review article on gaze typing systems and related research published thus far. Only small parts of this base paper have been directly reused in this thesis (found in several sections throughout the thesis), because many of the issues have been dealt with in more detail in later papers:

Majoranta, P. & Rähkä, K.-J. (2002) **Twenty years of eye typing: Systems and design issues**. *Proceedings of the Eye Tracking Research and Applications Symposium (ETRA '02)*, 15–22. New York: ACM Press. DOI: 10.1145/507072.507076. © 2002 ACM. Reprinted with permission.

I conducted the literature research for this review paper and wrote the first full version of it. Kari-Jouko Rähkä provided comments and wrote a few clarifying sentences.

Chapters 2–4 provide the background and an overview of eye tracking (Chapter 2), some information on gaze input (Chapter 3), and an introduction to issues involved in using the eye tracker as an assistive device (Chapter 4). Most of the text found in these chapters originates from this book chapter:

Majoranta, P., Bates, R., & Donegan, M. (2009a) **Eye-tracking**. In Constantine Stephanidis (ed.), *The Universal Access Handbook*, 587–606. Human Factors and Ergonomics series by Lawrence Erlbaum Associates, Inc. © 2009 by Taylor & Francis Group LLC - Books.

Reproduced with the permission of Taylor & Francis Group LLC - Books in the format Dissertation via Copyright Clearance Center.

I conducted the literature research and wrote the first full version of this overview article. Richard Bates corrected the language and wrote a couple of paragraphs and a few clarifying sentences. Mick Donegan contributed to the portions concerning end users (including a summary of user requirements and results from user trials – these parts were omitted or shortened for this thesis).

Chapter 5 introduces different methods for text entry by gaze and reviews related research. The text is partly based on the following book chapter, but it has been extended considerably and information on new systems (which appeared after it was originally published) has been added.

Majaranta, P. & Rähä, K.-J. (2007) **Text entry by gaze: Utilizing eye-tracking**. In I. S. MacKenzie & K. Tanaka-Ishii (eds.), *Text Entry Systems: Mobility, Accessibility, Universality*, 175–187. San Francisco: Morgan Kaufmann. © 2007. Text extracts and figures reprinted with permission from Morgan Kaufmann.

I conducted the literature research for this review paper and wrote the first full version of it. Kari-Jouko Rähä wrote the section on results of experiments (which part has been largely rewritten and moved to other sections of this thesis) and edited the language and grammar for the whole paper.

Chapter 6 introduces character and word prediction methods used in gaze-based text entry systems. This chapter was written specifically for this thesis and is not based on any papers I published previously.

Chapter 7 discusses interface design and the layout of the on-screen keyboard. It also summarizes the results of an experiment on a scrollable keyboard that can save screen space. This chapter is largely based on the following two papers. The first paper reports initial results from the first experiment, and the second paper extends the research with consideration of a follow-up experiment (only briefly summarized in this thesis) and reports full results from both experiments:

Špakov, O. & Majaranta, P. (2008) **Scrollable keyboards for eye typing**. *Proceedings of the 4th Conference on Communication by Gaze Interaction (COGAIN 2008)*, 63–66. Prague: CTU Publishing House (ISBN 978-80-01-04151-2). Available at <http://www.cogain.org/cogain2008/COGAIN2008-Proceedings.pdf>. © Špakov & Majaranta, with the COGAIN Network of Excellence. Reprinted with permission.

Špakov, O. & Majaranta, P. (2009, in press) **Scrollable keyboards for casual eye typing**. To appear in *PsychNology Journal*, in a special issue on gaze control for work and play. © Špakov & Majaranta, with *PsychNology Journal* (<http://www.psychnology.org/259.php>).

Oleg Špakov designed the scrollable keyboards and conducted the actual research. I was involved in writing the paper. The portion on related research has been extended for this thesis, and the results section has been abbreviated such that it only provides a summary of the results on the level appropriate for this thesis.

Chapter 8 reports results from three experiments studying the effects of feedback on gaze typing with varying dwell times. The chapter is largely based on this journal article:

Majaranta, P., MacKenzie, I. S., Aula, A., & Rähä, K.-J. (2006) **Effects of feedback and dwell time on eye typing speed and accuracy**. *Universal Access in the Information Society*, 5(2), 199–208. DOI: 10.1007/s10209-006-0034-z. © 2006 Springer. Reprinted with kind permission from Springer Science+Business Media.

I designed the experiments in consultation with the co-authors. I implemented the experimental software (including logging features) and ran the tests. For analysis, I received much help from Scott MacKenzie and Anne Aula, especially for the statistical tests. MacKenzie modified his Java program for text entry analysis so that it could be used to read the log files produced by the experimental software. I wrote the first version of the paper, and all co-authors contributed by commenting, writing some new text, and editing the text.

Preliminary results of the three experiments summarized in the aforementioned journal article were first published in the following conference papers. Parts of these articles have been included in *Chapter 8*, as additional details or discussion.

Majaranta, P., MacKenzie, I. S., Aula, A., & Rähä, K.-J. (2003a) **Auditory and visual feedback during eye typing**. In *Extended Abstracts of the ACM Conference on Human Factors in Computing Systems (CHI '03)*, 766–767. New York: ACM Press. DOI: 10.1145/765891.765979. © 2003 ACM, Inc. Reprinted with permission.

Majaranta, P., MacKenzie, I. S., & Rähä, K.-J. (2003b) **Using motion to guide the focus of gaze during eye typing**. *Abstracts of the 12th European Conference on Eye Movements (ECEM 12)*, O42. University of Dundee.

Majaranta, P., Aula, A., & Riih , K.-J. (2004) **Effects of feedback on eye typing with a short dwell time**. *Proceedings of Eye Tracking Research & Applications (ETRA '04)*, 139-146. New York: ACM Press. DOI: 10.1145/968363.968390.   2004 ACM, Inc. Reprinted with permission.

The contributions for all three of these papers are basically of the same nature: I designed the experiments in consultation with the co-authors. I implemented the experimental software (including logging features) and ran the tests. For analysis, I received a great deal of help from the co-authors, especially with the statistical tests. I wrote the first version of the papers, and all co-authors contributed with comments, by writing some new text, and by editing the text.

Chapter 9 introduces research related to studying how people learn to write by gaze, and it reports results from two experiments. The first experiment, reported in Section 9.2, is based on this paper:

Tuisku, O., Majaranta, P., Isokoski, P., & Riih , K.-J. (2008) **Now Dasher! Dash away! Longitudinal study of fast text entry by eye gaze**. *Proceedings of Eye Tracking Research & Applications (ETRA '08)*, 19-26. New York: ACM Press. DOI: 10.1145/1344471.1344476.   2008 ACM, Inc. Reprinted with permission.

I acted as an adviser for Outi Tuisku, who did her master's thesis work on this topic. She ran the experiments and analyzed the results with my help and supervision (and that of other co-authors). Tuisku and I wrote the paper together, each contributing about half of the text. She wrote the sections on method and results, while I wrote the introduction, material on previous research, discussion, and conclusions.

The second experiment, reported upon in Section 9.3, is based on this paper:

Majaranta, P., Ahola, U.-K., &  pakov, O. (2009b) **Fast gaze typing with an adjustable dwell time**. *Proceedings of the 27th International Conference on Human Factors in Computing Systems (CHI '09)*, 357-360. New York: ACM Press. DOI: 10.1145/1518701.1518758.   2009 ACM, Inc. Reprinted with permission.

As with the first experiment, I acted as the adviser, here for Ulla-Kaija Ahola, who conducted this research as her master's thesis work. She ran the experiments and analyzed the results under my supervision. Oleg  pakov implemented the experimental software. I wrote the paper, on which Ahola and  pakov offered comments.

Chapter 10 outlines directions for future research and reports preliminary results from an initial experiment studying the usability of a gaze-operated dynamic pie menu for text editing by gaze:

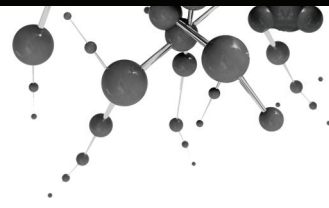
Majaranta, P., Majaranta, N., Daunys, G., & Špakov, O. (2009c, in press) **Text editing by gaze**. *Proceedings of the 5th Conference on Communication by Gaze Interaction (COGAIN 2009)*, 19–23. IMM-Technical Report, Technical University of Denmark (ISBN 978-87-643-0475-6). Available at <http://www.cogain.org/cogain2009/COGAIN2009-Proceedings.pdf>. © Majaranta, Majaranta, Daunys, & Špakov, with the COGAIN Network of Excellence. Reprinted with permission.

The experimental design and the requirements for the experimental software were specified jointly by all authors. Gintautas Daunys implemented the experimental software. Niina Majaranta conducted the user trials and performed initial analysis of the results as part of her bachelor's thesis work. I analyzed the results further and wrote the first version of the paper.

Contents

1	INTRODUCTION	1
1.1	Objective.....	1
1.2	Methods and Measures.....	2
1.3	Results	3
2	EYE TRACKING.....	5
2.1	The History of Eye Tracking	5
2.2	Contemporary Technologies.....	7
3	BASICS OF GAZE INPUT	13
3.1	The Nature of Eye Movements	13
3.2	Calibration	14
3.3	Accuracy Limitations	16
3.4	Gaze Pointing.....	16
3.5	Selection Techniques	17
4	THE EYE TRACKER AS AN ASSISTIVE DEVICE	21
4.1	Communication and Control	21
4.2	Assistive Applications of Eye Tracking.....	26
5	TEXT ENTRY BY GAZE	33
5.1	Text Entry by Direct Gaze Pointing	33
5.2	Text Entry through Eye Switches	36
5.3	Text Entry by Discrete Gaze Gestures	37
5.4	Text Entry by Continuous Pointing Gestures.....	44
6	CHARACTER AND WORD PREDICTION.....	49
6.1	In Search of Better Typing Speed	49
6.2	Predicted Word Lists.....	50
6.3	Character Prediction.....	52
6.4	The Cost of the Additional Cognitive and Perceptual Load	53
6.5	Further Reading	55
7	LAYOUT.....	57
7.1	Coping with Inaccurate Tracking.....	57
7.2	Saving Screen Space with Compact Keyboard Layouts.....	59
7.3	Scrollable Keyboards.....	60
8	FEEDBACK	69
8.1	Related Research.....	70
8.2	Methods and Procedures.....	73
8.3	Effects of Auditory and Visual Feedback.....	75
8.4	Effects of Animated Feedback	79
8.5	Effects of Feedback with a Short Dwell Time	82
8.6	Discussion.....	88
8.7	Guidelines	91
8.8	Conclusion	93

9	LEARNING TO WRITE BY GAZE	95
9.1	Learning Voluntary Gaze Control.....	95
9.2	Learning to Write by Gaze via Continuous Gestures.....	98
9.3	Learning to Type by Gaze with an Adjustable Dwell Time	115
10	MOVING FROM TEXT ENTRY TO EDITING BY GAZE	129
10.1	Editing Text by Gaze	129
10.2	Future Research: Involving Users with Disabilities.....	136
11	SUMMARY AND CONCLUSIONS	139
12	REFERENCES	143



1 Introduction

1.1 OBJECTIVE

The aim of this thesis is to present a comprehensive study of the gaze-based text entry process and to find ways to make the interaction more efficient and enjoyable.

This thesis examines the voluntary use of human eye movements as an input method for communication and control of a computer by gaze. The use of natural eye movements in gaze-aware and attentive systems (see, e.g., Hyrskykari et al., 2005; Vertegaal, 2003) is beyond the scope of the thesis. General features and methods of gaze input are introduced briefly, but the bulk of the thesis concentrates on studying various aspects of text entry by gaze alone.

Text entry by gaze is used by people with severe disabilities, for whom eye movements may be the only means of communication available. The basic methods for producing text by gaze have been studied and have been in real-world use since the early 1980s (ten Kate et al., 1979; Levine, 1981; Friedman et al., 1982; Hutchinson et al., 1989). However, the design issues have not been studied in detail until recent years (Istance et al., 1996; Hansen et al., 2001; Majaranta & R ih a, 2002, 2007). Text entry by gaze involves a rich set of issues for study both from the practical standpoint – *for development of more usable systems* – and from the research point of view: *to better understand the properties of gaze in communication and text entry tasks.*

The main research questions addressed in this thesis are:

- What kinds of processes are involved in gaze-based text entry, and how could we improve its usability through better interface design?
- What are the effects of auditory and visual feedback on gaze typing performance, and how could proper feedback facilitate the tedious task of text entry by gaze? The question of visual feedback is especially interesting, since the same modality (vision) is used for both input and output.
- How long does it take for novices to learn gaze typing, and how quickly can they enter text by gaze alone?
- How could special interaction widgets assist in text entry and text editing by gaze?

1.2 METHODS AND MEASURES

To gain insight into the task of text entry by gaze, a thorough literature review and survey of existing gaze typing systems was carried out.

Prototypes of a simple gaze typing system were constructed and used in several user trials. The prototypes included log features for saving gaze and event data.

Controlled experiments and longitudinal user trials were conducted to study various interface design issues, such as the effect of feedback and dwell time duration.

The analysis of user performance included typing speed, accuracy, gaze behavior, and responses from the interviews.

Typing speed was measured in words per minute (wpm), where a word is any sequence of five characters, including letters, spaces, punctuation, etc. (MacKenzie, 2003).

In measurement of *accuracy*, both corrected errors and errors left in the final text were taken into account. The metrics used were error rate and keystrokes per character.

Error rate was calculated by comparing the transcribed text (text written by the participant) with the presented text, using the minimum string distance (MSD) method described by Soukoreff and MacKenzie (2001, 2003). This method does not take into account corrected errors.

Keystrokes per character (KSPC) (MacKenzie, 2002; Soukoreff & MacKenzie, 2003) is a measure of the average number of keystrokes used to enter each character of text. Ideally, $KSPC = 1.00$, indicating that each key press produces a character. If participants correct mistakes during entry, the KSPC value is greater than 1. For example, if “hello” is entered as h e l x [del] l o, the final result is correct (0% error rate), but the KSPC value is $7 / 5 = 1.4$ (seven keystrokes for entering five characters). KSPC is an accuracy measurement reflecting the overhead incurred in correcting mistakes.

In addition to typing speed and accuracy, various aspects of gaze behavior were studied.

Read text events (RTE) refers to a participant switching the point of gaze from the virtual keyboard to the typed text field to review the text written so far. Instead of reporting raw counts, RTE is normalized and reported on a per-character basis. The ideal value is 0, implying that participants were confident enough to proceed expeditiously without verifying the transcribed text. Typically, however, participants occasionally review their work. This is known to occur more frequently for inexperienced participants (Bates, 2002); however, the type of feedback may also have an effect, as seen in the results of the experiments, discussed below.

Re-focus events (RFE) is a measure of the average number of times a participant re-focuses on a key to select it. As with read text events, the RFE value is normalized and reported on a per-character basis. RFE is ideally 0, implying that the participant focused on each key just once. If the participant’s point of gaze leaves a key before it is selected, and then re-focuses on it without selecting anything else in between, RFE is greater than 0.

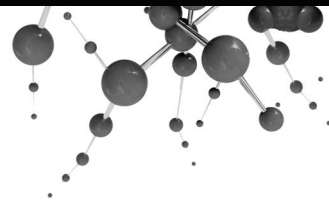
In addition, participants were observed during the experiments, and their subjective impressions and preferences were collected by means of questionnaires and interviews.

1.3 RESULTS

As a result of the literature review, this thesis provides an extensive overview of the research conducted in the area of gaze-based text entry. It is hoped that the thesis will be a useful starting point and resource for researchers or assistive technology professionals wishing to learn more about gaze-based text entry and will provide useful guidelines for developers of gaze typing systems.

Results from the experiments show that *small improvements in the interface design can lead to significantly improved user performance and satisfaction*. For example:

- A customized interface and layout can improve user performance and support the varying needs of the user in general, or in certain situations. For example, a full on-screen keyboard enables fast text entry by direct gaze pointing, but it takes a lot of space. A solution is to use a scrollable keyboard, which can save screen space but is still immediately usable since it preserves the familiar layout of the keys.
- Proper feedback significantly improves user performance and satisfaction during gaze typing. For example, adding a simple “click” to confirm selection can improve the text entry speed over that seen with plain visual feedback. The improvement is small, but the effect accumulates in the repetitive task of text entry.
- The possibility of adjusting the dwell time duration supports learning and enables fast text entry with an on-screen keyboard.
- Preliminary results indicate that special gaze-operated widgets may enhance text editing in certain situations and could provide a useful alternative to menu-based commands.



2 Eye Tracking

2.1 THE HISTORY OF EYE TRACKING

Early in the development of the field of eye gaze tracking, eye movements were studied mainly to observe the nature of human eye movements, rather than to use these movements for communication. The first *eye tracking devices* that produced objective and accurate data were highly invasive and uncomfortable. For example, the system developed by Delabarre in the late 1800s used an eye cup with a lever extending to draw the eye movements on a smoked drum. The eye cup was attached directly to the surface of the eye (which required anesthetization with cocaine) and had a hole in it through which the test subject could see (Wade & Tatler, 2005). A breakthrough in eye movement research was the later development of the first “non-invasive” eye tracking apparatus by Dodge and Cline in the early 1900s (Wade & Tatler, 2005). This was based on photography and light reflected from the cornea (the shiny reflective surface of the eye). Many basic properties and types of eye movements were categorized via the camera-based device of Dodge and Cline or later, improved versions. The “Dodge Photochronograph” is seen as the inspiration to, and first ancestor of, the current video-based, corneal reflection eye tracking systems discussed later in this work.

The development of computing power enabled gathering of eye tracking data in real time, as well as the development of assistive technology systems aimed directly at people with disabilities – for example, ten Kate et al., 1979; Levine, 1981; Friedman et al., 1982; Yamada & Fukuda, 1987; Hutchinson et al., 1989, all of whom indeed focused primarily on users with disabilities. These first systems were typically based on *eye typing*, or *gaze typing*, where

the user could produce text by using the focus of gaze as a means of input. One of the earliest eye typing systems, the Eye-Letter-Selector (ten Kate et al., 1979), is shown in Figure 2.1. Here eye movements were detected by two phototransistors attached to eyeglass frames (the frames are located on top of the device in Figure 2.1).

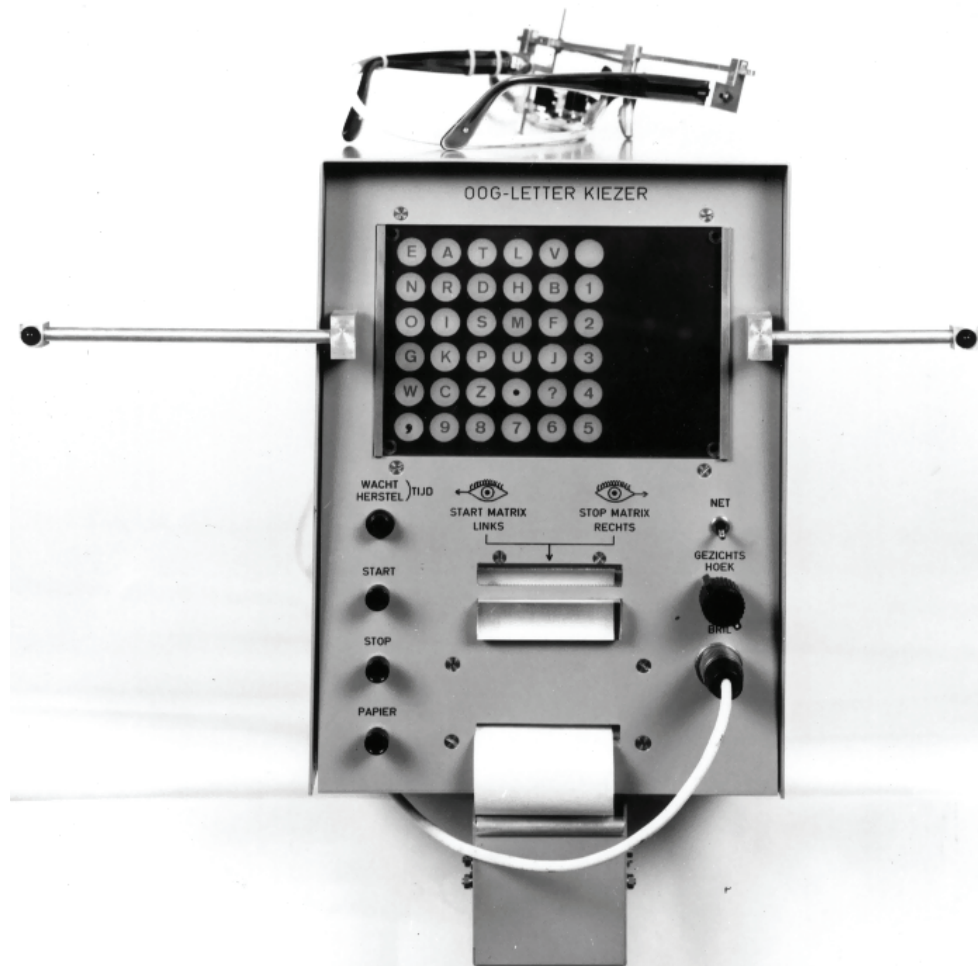


Figure 2.1. The Eye-Letter-Selector[®] detected rough horizontal eye movements¹ (photo courtesy of Dr. ir. E. E. E. Frietman).

The Eye-Letter-Selector could not track eye gaze sufficiently accurately to allow direct selection of individual characters on the keyboard. Instead, it detected eye movements to the left or right and used these as a single or double eye-controlled switch system (ten Kate et al., 1979). To enable typing, the system adopted a column-row scanning procedure (illustrated in Figure 5.3). When the scanning reached the column, and subsequently

¹ A detailed, illustrated description of the system and its later variations is available at <http://www.ph.tn.tudelft.nl/~ed/ELS-Handi.html> (accessed 1 March 2009).

the row where the desired letter was, the user selected it with the eye switch by looking right. This enabled slow but effective eye typing.

2.2 CONTEMPORARY TECHNOLOGIES

Current eye tracking technologies have evolved from early systems such as the Eye-Letter-Selector into a range of technologies: electro-oculography (EOG), where the user wears small electrodes around the eye to detect the eye position (Figure 2.2); the scleral contact lens / search coil system, in which the user wears a contact lens with a magnetic coil on the eye that is tracked by external magnetic systems; video-oculography (VOG) or photo-oculography (POG), where still or moving images are taken of the eye to determine the eye's position; and finally video-based combined pupil/corneal reflection techniques that extend VOG by artificially illuminating both the pupil and cornea of the eye (Figure 2.3) for increased tracking accuracy (Duchowski, 2003).

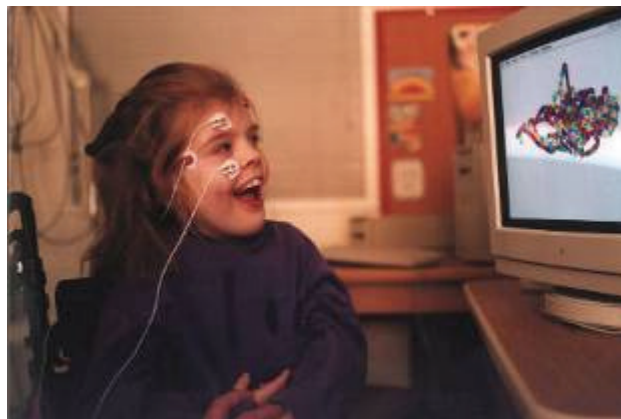


Figure 2.2. Eye painting with EagleEyes (<http://www.eagleeyes.org>) (photo courtesy of Dr. James Gips).

With each of these approaches taken in turn, EOG-based systems may be seen as impractical for everyday use, because they require electrodes to be placed around the eye to measure the skin's electrical potential differences. There are, however, EOG systems that are successfully used for augmentative and alternative communication (see, for example, Gips et al., 1993; Hori et al., 2006). For example, the EagleEyes system (Gips et al., 1993) has improved the quality of life of numerous users (see Figure 2.2). There are still drawbacks, since some people may not wish to have electrodes placed on their face, and the electrodes can fall off if the user perspires (Betke et al., 2002). Systems based on electro-oculography are not, however, sensitive to changes in lighting conditions (especially outdoor lighting), which pose a considerable problem for video-based systems.

As the EOG potential is proportional to the angle of the eye in the head, an EOG-based mouse pointer is moved by changing the angle of the eyes in the head (EagleEyes, 2000). The user can move the mouse cursor by moving the eyes, the head, or both. More information about the EOG-based EagleEyes system is available in the work of DiMattia et al. (2001) or at <http://www.eagleeyes.org/>.

Systems that use contact lenses or in-eye magnetic coils are mainly used for psychological or physiological studies that require high accuracy (these systems can be very accurate, to a fraction of a degree). Here gaze tracking is used as an objective and quantitative method of recording the viewer's point of regard. Such information can be used for medical and psychological research to gain insight into human behavior and perception (see, e.g., Rayner, 1995).

Video-oculography and photo-oculography camera-based systems are considered to be the least obtrusive and thus are the means best suited to interactive applications that react to the user's gaze at some level (Morimoto & Mimica, 2005). These systems tend to be inaccurate so are enhanced by means of pupil detection combined with corneal reflection to provide a point of regard (POR) measurement, which means that the system can calculate the direction of gaze (Duchowski, 2003).

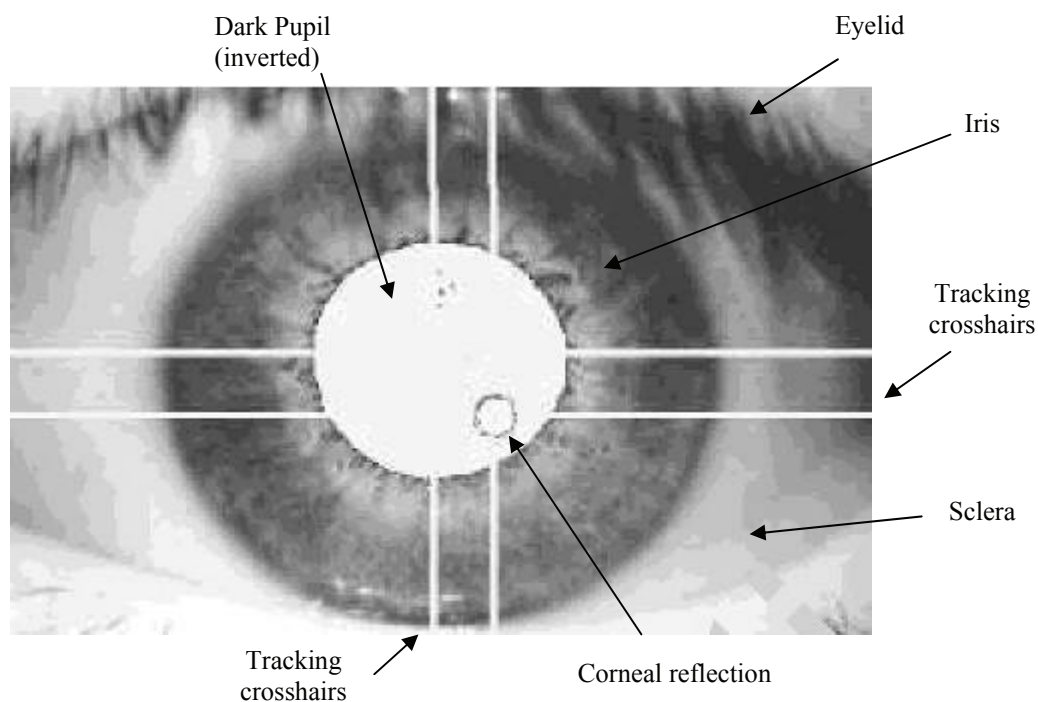


Figure 2.3. Video frame from a VOG system showing eye corneal reflection and pupil detection (image courtesy of Dr. Richard Bates).

In practice, at least two reference points are required for gaze point calculation. By measuring the corneal reflection(s) from an infrared artificial light source aimed on- or off-axis at the eye relative to the center of the pupil, the system can compensate for inaccuracies and also for a limited degree of head movement. Gaze direction in these systems is calculated by measuring the changing relationship between the moving bright (if the light is aimed on-axis) or dark (light aimed off-axis) pupil of the eye and the essentially static reflection of the infrared light source back from the surface of the cornea. This approach relies on shining infrared light (to avoid the tracked subject squinting) at an angle onto the cornea of the eye, with the cornea producing a reflection of the illumination source (see Figure 2.3).

In practice, the corneal reflection remains roughly constant in position during eye movement, hence the reflection will remain static during rotation of the eye and changes in gaze direction, thus giving a basic eye and head position reference. This reflection also provides a simple reference point to compare with the moving pupil, and so enables calculation of the gaze direction vector of the eye (for a more detailed explanation, see Duchowski & Vertegaal, 2000).

Most of the currently available eye control systems are video-based (VOG) with corneal reflection; therefore, this chapter concentrates mostly on these video-based systems. For a detailed survey of techniques for eye detection, eye tracking, and gaze estimation, see Hansen and Ji (2009).

Only a minority of the tens of currently available eye tracking systems are targeted at people with disabilities. Most of the systems use the same basic technical principles of operation, but what makes certain systems suitable for people with disabilities are the applications (software) that are supported or come with the system and the (technical) support and accessories provided by the manufacturers and retailers. For a disabled person, an eye control system is a way of communicating and interacting with the world and may be used extensively, daily and in varying conditions. Thus, reliability, robustness, safety, and mounting issues must be carefully taken into account, in addition to ease of use and general usability. Table 2.1 lists commercially available gaze communication systems targeted at people with various (dis)abilities and needs. These systems are used as assistive devices. Eye tracking systems used for general research and analysis are not listed here.²

² For a list of systems for research and analysis, see <http://www.cogain.org/eyetrackers/eyetrackers-for-eye-movement-research> (accessed 1 March 2009).

Table 2.1: Commercially available (video-based) eye-control systems³

<p>Alea Technologies: Intelligaze IG-30 (http://www.alea-technologies.com/)</p> <p>Actual functionality that depends on the installed (AAC) applications. The IG-30 system acts as a gaze input device, which can then control the Windows desktop, many standard applications, or dedicated elements. The eye tracking module can be integrated with different screen sizes or a desktop PC that can be mounted to a wheelchair.</p>
<p>DynaVox Technologies: EyeMax (http://www.dynavoxtech.com/)</p> <p>System based on EyeTech's gaze tracking technology. EyeMax makes the eye tracking access method available to communicators who use DynaVox Vmax. It comprises two parts: DynaVox Vmax and the DynaVox EyeMax Accessory. The EyeMax system allows augmented communicators to access their Vmax with a simple blink, via switch selection, or by dwelling on the desired area of the screen. Available in several languages and fully portable, with its own internal batteries.</p>
<p>Eye Response Technologies: ERICA (http://www.eyerresponse.com/)</p> <p>Via mouse emulation, full control of Windows (typing, e-mail, Web browsing, games, etc.). Portable, with flexible mounting options. Environmental control and remote IR control available as accessories. Touchscreen and head control also possible. Comes with desktop or laptop PC, available for Windows and Macintosh.</p>
<p>LC Technologies: Eyegaze (http://www.eyegaze.com/)</p> <p>Dedicated, eye-controlled keyboard and phrases, allowing quick communication and synthesized speech. Access to the Internet and e-mail. One can play eye-controlled games (included), run computer software, and operate a computer mouse. Includes also support for book reading, and control of lights and appliances (environmental control).</p>

See also Oleg Špakov (2008) *iComponent - Device-Independent Platform for Analyzing Eye Movement Data and Developing Eye-Based Applications*. Dissertations in Interactive Technology, Number 9. University of Tampere. Available online at <http://acta.uta.fi/teos.php?id=11064> (accessed 1 March 2009).

³ For an up-to-date list of eye tracking systems used as assistive devices, see <http://www.cogain.org/eyetrackers/> (accessed 1 March 2009).

Tobii Technology: MyTobii

(<http://www.tobii.com/>)

Dedicated eye typing, e-mail, and gaze-controlled games included. Includes mouse emulation that can be used to control Windows. Dwell time, a switch, or a blink can be used to click. Tracks both eyes. Good tolerance for head movements. Long-lasting calibration with minor drifting. Accessories include a mounting arm. Available in several languages.

EyeTech Digital Systems: EyeTech TM3

(<http://www.eyetechds.com/>)

Mouse emulation, allowing full control of Windows (typing, e-mail, Web browsing, games, etc.). A switch or blink can be used to click, in addition to dwell time selection. Several models are available, with varying properties. Allows moderate head movements. Portable and comes with a tablet PC. The tracking module can also be purchased separately. Available in several languages.

Metrovision: VISIOBOARD

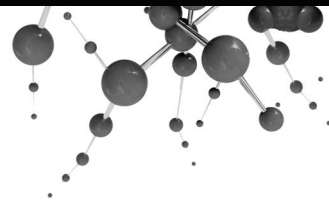
(<http://www.metrovision.fr/>)

Mouse emulation enabling full control of Windows (typing, e-mail, Web browsing, games, etc.). Clicking can be done via dwell time (staring), blinks, or an external switch. Allows moderate head movements. Mounting arm for people in seated or lying position (cannot be attached to a wheelchair).

H.K. EyeCan: VisionKey

(<http://www.eyecan.ca/>)

Head-mounted, lightweight, fully portable eye communication system. Comes with a standalone, separate control unit with a display (attached to eyeglass frames) and a voice synthesizer, so no computer is needed when the user is on the move. A standard USB keyboard interface is provided for computer control. Compatible with Windows or Macintosh. Provides scanning options for people with limited eye control. Independent of (non-violent) head/body movements.



3 Basics of Gaze Input

3.1 THE NATURE OF EYE MOVEMENTS

It is known that we look at things by holding our gaze relatively still⁴ on an object for a short while, long enough for the human brain to perceive the nature of the object. Such a *fixation* typically lasts approximately 200–600 ms. Between fixations, gaze jumps rapidly from one object to another, with these *saccades* typically lasting approximately 30–120 ms each (Jacob, 1995). Saccades are ballistic movements; once a saccadic jump has been started, it cannot be interrupted, nor can its direction be changed. In addition to making saccadic movements, the eyes can smoothly follow a moving target; this is known as (smooth) *pursuit*. Normal eye movement is thus composed of fixations on objects of interest joined by rapid saccades between those objects, with occasional smooth pursuit of moving objects.

Examining the retina, one can see that the size of the high-acuity field of vision, the *fovea*, gives accurate vision that subtends an angle of about one degree from the eye. To illustrate this approximately, the diameter of the high-acuity circular region corresponds to an area of about two degrees, which is about the size of a thumbnail when viewed with the arm extended (Duchowski & Vertegaal, 2000). Everything inside the foveal

⁴ The eyes make very small, rapid movements even during fixations to keep the nerve cells in the retina active and to correct slight drifting in focus. These “tremors” and “microsaccades” are so small that they are of little importance for practical applications of eye tracking.

area is seen in detail, with everything outside this narrow field seen indistinctly. Thus, people see only a small portion of any full scene in front of them accurately at a time – it is this narrow vision that generates the need to move the eyes rapidly around to form a full view of the world. The further away from the fovea an object is, the less detailed it appears to the human eye. The remaining peripheral vision provides cues about where to look next and also gives information on movement or changes in the scene in front of the viewer (for more information about eye movements and visual perception, see, for example, Haber & Hershenson, 1973).

Since the foveal area of visual acuity is fairly small, and since people actually need to direct their gaze almost directly towards the object of interest to get an accurate view of it (within one degree or so), tracking of gaze direction becomes possible – the user is probably looking at and perceiving the object being pointed at by the eyes.

3.2 CALIBRATION

Before a video-oculography eye tracking system can calculate the direction of gaze, it must be calibrated for the specific user. This is usually done by showing a few (for example, nine equally spaced) points on the screen and asking the user to gaze at the points, one at a time (see Figure 3.1). The images of the eye are analyzed by the computer, and each image is associated with corresponding screen coordinates. These main points are used to calculate any other point on-screen via interpolation of the data. The accuracy of such systems is very much dependent on successful calibration.

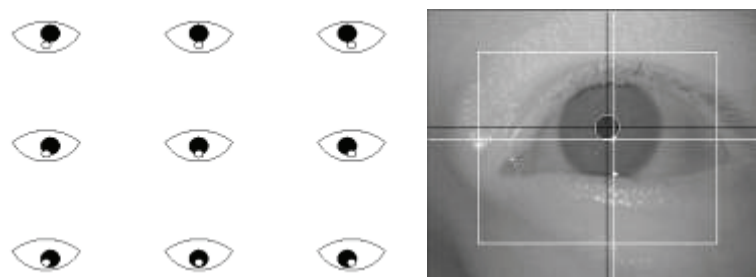


Figure 3.1. An illustration of pupil movements (black circles) and nearly stationary corneal reflections (smaller white circles) as seen by an eye tracker’s camera at each of the nine calibration points (on the left) and an image taken by an eye tracker’s video camera (on the right). Note how the relationship between pupil and corneal reflection changes as the eye gaze direction changes.

Most current eye tracking devices achieve an accuracy of a 0.5-degree visual angle from the user (this is the equivalent of a region of approximately 15 pixels on a 17-inch display with a resolution of 1024 x 768 pixels viewed from a distance of 70 cm). The practical accuracy of the

system may be less because of “drifting,” where over time the measured point of gaze drifts away from the actual point of gaze. This drift is caused by the changes in the characteristics of the eyes and is mainly due to changes in pupil size, and compensation errors from excessive movement of the head that results in the eye moving away from the clear view of the camera and the original calibration position (Tobii, 2006). The effects of drifting can be taken into account and dynamically corrected to some extent (Stampe & Reingold, 1995). Inaccuracy in pointing is corrected by realigning the possibly inaccurate measured gaze position with the center of any object selected. It is (often correctly) assumed that the user is looking at the center of the object he wishes to select. Thus, if the measured point of gaze does not match the coordinates at the center of the object, it is possible to correct the drift by realigning the measured gaze position to the center of the object – where the user is most probably looking – thus correcting the drift. Such automatic drift correction is done dynamically while the user is using the software. Obviously, the automatic drift correction does not work if the calibration is too far off. It is, therefore, important to provide the possibility for easy recalibration at any time.

Some eye tracking systems have additional techniques for preventing drifting. Using data from both eyes may help, as the system may continue with data from one eye if the other is lost. For example, Tobii trackers use averaging of data from both eyes to minimize the drifting effects (Tobii, 2006). This “binocular averaging” enables long-lasting calibration with very little drifting and saves the user from continuous recalibration.

A VOG eye tracker must have an unobstructed view of the eye and pupil if it is to be able to track the eye. Eyelids or lashes may partially cover the pupil, and ambient light or reflections from the environment may cause problems. Eyeglass lenses or frames may cause extra reflections, and when contact lenses are used, the reflection is obtained from the surface of the contact lens instead of the cornea. This can cause problems if the lenses are displaced over time, causing degradation in tracking accuracy. Problems may be prevented or minimized through careful setup – for example, by minimizing changes in the lighting conditions and positioning the camera such that it has a clear view of the user’s eye (Goldberg & Wichansky, 2003).

Finally, most eye tracking systems have problems in the event of severe involuntary head or eye movements. Certain medical conditions may also prevent successful calibration (Donegan et al., 2005). In some cases, calibration may be totally impossible or very inaccurate. If the calibration fails, some systems can be used with a default calibration, and special filtering of eye movements can be applied if the user has eye movement disorders (Charlier et al., 1997).

3.3 ACCURACY LIMITATIONS

The accuracy of the measured point of gaze is a problem if a user wishes to use gaze as the main method to control a standard graphical computer interface. Many of the target objects in typical graphical user interfaces are smaller than the area of high-acuity vision (such tiny objects subtend an angle of less than one degree at a normal viewing distance from the screen). Even if eye trackers were perfectly accurate, the size of the fovea would restrict the practical accuracy of the systems. Everything inside the foveal region is seen in detail without movement of the eye. Also, attention can be retargeted within the foveal region at will without actually moving the eyes, making it practically impossible to determine the exact pixel the user is looking at on the screen. Thus, gaze is not as accurate an input device as devices such as a desktop hand mouse, but it can be much faster at pointing, because of the speed of the eye, if the target objects on the screen are large enough (Sibert & Jacob, 2000; Ware & Mikaelian, 1987).

Increasing the size of the targets on the screen makes them easier to “hit” and improves the performance of eye gaze input. This results in objects designed for eye gaze control often being quite large on the screen. Since fewer keys are shown at a time, they must be organized hierarchically in menus and sub-menus. This slows down the gaze interaction; however, it is important to acknowledge that eye control can still be an option worth considering even with very poor calibration. Making on-screen objects much larger can make the difference between a user being able to use an eye tracking device or not being able to use it at all (Donegan et al., 2005).

3.4 GAZE POINTING

Eye movements are so rapid that it is not always easy to realize how much and how often the eye moves. Gaze is easily attracted (or distracted) by movement in the peripheral vision, resulting in unwanted “flicks” away from objects of interest. Eye movements are also largely unconscious and automatic; people do not normally need to think about where to look. When necessary, however, one can control gaze at will, which makes eye control possible.

Gaze pointing, or placing the computer mouse cursor where the user is looking on the computer screen, is an intuitive method that requires little training (Stampe & Reingold, 1995), since it mimics the operation of a normal desktop mouse. However, it should be noted that, for a profoundly disabled person who does not have prior experience of any method of computer control, it may take time to master a gaze pointing eye control system (Donegan et al., 2006b; Gips et al., 1996).

3.5 SELECTION TECHNIQUES

Because the same modality, gaze, is used for both perception (viewing the information and objects on a computer screen) and control (manipulating those objects by gaze), a gaze-based communication system should be able to distinguish casual viewing from the desire to produce intentional commands. This way, the system can avoid the “Midas touch” problem (Jacob, 1991), wherein all objects viewed are unintentionally selected. The obvious solution is to combine gaze pointing with some other modality for selection. If the person is able to produce a separate “click,” then this click can be used to select the item in focus. This can be a separate switch, a blink, a wink, a sip, a puff, a wrinkling of the forehead, or even smiling or any other muscle activity available to the user (Barreto et al., 2000; Fono & Vertegaal, 2005; Huckauf & Urbina, 2008b; Junker & Hansen, 2006; Kumar et al., 2007; Monden et al., 2005; Surakka et al., 2003; Surakka et al., 2004; Ware & Mikaelian, 1987). In addition, blinks and winks can be detected from the same video signal used to analyze eye movements, removing the need for additional switch equipment. Since muscle activity may be extremely faint or weak, it is typically measured via electromyography, EMG, of any available working muscles. Some systems are based solely on blinks or winks (using the eyes as a kind of switch), without tracking of gaze direction. For more information about such systems, see, for example, Grauman et al. (2003) or Murphy and Basili (1993). Since people blink naturally several times per minute, so an intentional blink needs to be longer than an automatic blink (i.e., it must last longer than 300–400 ms, according to Huckauf & Urbina, 2008b). Gaze can also be combined with speech (Kaur et al., 2003; Miniotas et al., 2006). However, those who need gaze communication most are often unable to speak, so, obviously, verbal communication is not an option for them.

If a user is capable of moving only the eyes or has very limited other motor control, separate switches are not an option, and the system must be able to separate casual viewing from intentional eye control. The most common solution is to use *dwelt time*, prolonged gaze, with a duration longer than that of a typical fixation⁵ (typically, 500–1000 ms; see, for example, Hansen et al., 1995; Hansen et al., 2003a; Istance et al., 1996; Majaranta & Rähkä, 2002; Velichkovsky et al., 1997). Most current eye control systems provide adjustable dwell time as one of the parameters for the selection method. Requiring the user to fixate for a long time does

⁵ As an interesting side note, it can be mentioned that in manual pointing tasks (such as pointing with a hand at an object that is being referred to in speech) the natural dwell time during pointing to express “selection” of a target seems to be approximately 350–600 ms (Müller-Tomfelde, 2007).

reduce false selections, but it is uncomfortable for the user, since fixations longer than 800 ms are often broken by blinks or saccades (Stampe & Reingold, 1995). A long dwell time may also be tiring to the eyes and hinder concentration on the task (Majaranta et al., 2006).

Another solution to the Midas touch problem is to use a special selection area (Yamada & Fukuda, 1987) or an on-screen button (Ware & Mikaelian, 1987; Ohno, 1998). For example, in the “quick glance” method developed by Ohno (1998), each object that could be selected was divided into two areas: command name and selection area. Selection was done by first fixating briefly on the command (to determine the name or type of the command), then confirming that a selection was required, via a brief glance at the selection area. Alternatively, a user who is experienced and knows the locations of the commands need only glance directly at the selection area associated with that command.

Gaze gestures can also be used to make a selection by gaze alone. The user initiates a command by making a sequence of “eye strokes” in a certain order. Figure 3.2 shows an example gesture. A gesture does not require dwell time, though dwell can be used to start a gesture or separate several gestures from each other. Making a gaze gesture still requires a brief stop (fixation) between the strokes (saccades), each of which costs some time (each fixation costs from 150 to 600 ms, according to Duchowski, 2003). Therefore, it has been estimated that a gesture may consist of a maximum of four strokes if it is to be able to compete with (relatively long) dwell time (Huckauf & Urbina, 2008b; Wobbrock et al., 2008). The number of strokes needed for a gesture depends on the size of the alphabet (how many distinguishable gestures are needed). The implementation also has an effect: if the starting point and “hot spots” for the gesture are known, then, in principle, even one stroke is enough for a well-defined gesture. However, if the recognition of a gesture is made independently from the location, it needs to be complex enough to be clearly distinguishable from natural viewing patterns.

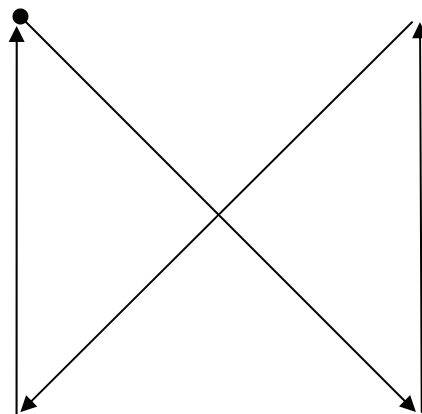


Figure 3.2. An example of a gaze gesture, starting from the top-left corner, with a gazing order of SE, N, SW, N.

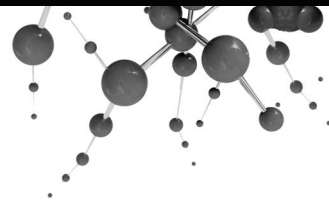
A gaze gesture can be independent of the screen location and only rely on relative changes in the direction of gaze, or it may rely on certain locations (hot spots or target areas) on or off the screen. For example, Isokoski (2000) used off-screen targets, at which the user had to gaze in a certain order to enter a command. By using off-screen targets, he was able to save the full screen for other purposes.

Milekic (2003) proposed gaze gestures for manipulation of art objects (e.g., pictures of paintings or sculptures); for example, a rapid gaze upwards would grab the object and a rapid glance downwards would drop it, thus performing a gaze-gesture-based drag and drop. Similarly, with a rapid glance sideward, the user could “throw” the item off the screen, which would bring up the next item in the art catalogue. Istance et al. (2008) used four simple off-screen glances to switch between modes. The gaze gestures enabled a quick and effortless change between different mouse operations: a quick glance down enabled the dragging mode, a glance to the left activated the left mouse click, a glance to the right parked the mouse in its current position (enabling free viewing of a separate menu and dwell time activation of a command at the pointer position), and a quick glance up turned off the eye control.

Gestures that are not bound to certain locations (on the screen) but are based on relative change in the direction of gaze are insensitive to accuracy problems and calibration shifting (Drewes & Schmidt, 2007). They are especially useful with small screens and in mobile situations involving mobile trackers. For example, gaze gestures can be used to control a mobile phone (Bulling et al., 2008a) or to play a game, using a lightweight wearable tracker (Bulling et al., 2008b). Obviously, gaze gestures that are used in free viewing situations should be complex enough to differ from natural gaze patterns but still simple enough that people can easily learn and remember them.

In their search for alternatives to dwell time, researchers have developed totally new approaches for selection by gaze. For example, Huckauf et al. (2005) used anti-saccades to solve the Midas touch problem. When performing an anti-saccade, the user voluntarily looks in the direction opposite where the saccade would naturally be directed. In practice, for this movement, one draws a copy of a screen object (button) next to the original object – let us say on the right side of that object. Naturally the human gaze would automatically look at this new object by making a saccade towards the new object. An anti-saccade requires the user to make a similar eye movement but to the other side of the original object – in this case, to the left side of the object (since the copy appeared on the right side). Anti-saccades can be faster than dwell times and faster even than a mouse click, but they produce significantly higher error rates (Huckauf, 2005).

Sometimes gaze is unfocused or undirected when the attention of the user is not directed at interaction with the screen. For example, a user may be concentrating or thinking about something, and during this thinking the eyes may wander about the screen and accidentally or without conscious intent point at an object. Since users' eyes are always "on" (they are always pointing somewhere unless closed), there is always a risk (and the annoyance and fear) of accidentally staring at an object that is then unintentionally selected. This results in the user feeling unable to relax fully. Thus, in addition to a long enough dwell time, it is beneficial to the user if eye control can be paused with, for example, an on-screen "pause" command, to allow free viewing of the screen without fear of the Midas touch (Donegan et al., 2006a).



4 The Eye Tracker As an Assistive Device

4.1 COMMUNICATION AND CONTROL

For people with severe motor disabilities, eye gaze may be the only communication option available. For example, after a severe accident, a person may not be able to speak, in which case a doctor may ask the person to “look up” or “look down” as an indication of understanding and agreement. This method of communication can be expanded from a simple “yes” or “no” command to a full communication system by adding meaningful objects to the view of the user. An example of this approach is the gaze communication board (see Figure 4.1). The board has pictures, commands, or letters attached to it, with the user selecting items on the board by looking at them. The person, or interpreter, on the other side of the transparent board interprets the message by following the eye movements of the user onto the differing targets. Such a system illustrates the simple communication power of eye gaze tracking. For more examples of low-tech means of gaze communication, see the work of Goossens’ and Crain (1987) or Scott (1998).

Manual gaze-based communication aids such as the “E-Tran Frame” are not always convenient, private, or practical, and they may not possess all communication functions a user may wish to use. Hence, computer-based gaze communication systems have been developed, wherein an eye tracking device and a computer replace the manual communication board. An eye tracking device records the eye movements and a computer

program analyzes and interprets them in place of the human operator. This forms a basic computer-aided gaze communication system.



Figure 4.1. A gaze communication board (“E-Tran frame”). The person on the other side of the board acts as a human eye tracker and interprets the direction of gaze through the transparent board. A letter is chosen by first looking at it and then looking at the color button that corresponds to the color of the letter.⁶

When an eye tracker is used as an assistive device, it provides a way of communicating for a person who cannot talk, and a way of interacting with the world for a person whose mobility is restricted. This section discusses ways of implementing the most common functions of eye control and provides a few examples of gaze-controlled applications, discussing special design issues that arise from using gaze input.

Mouse Emulation

As introduced earlier, one common way of implementing eye control is to use eye movements to control the mouse cursor. Binding eye movements

⁶ Downloadable gaze communication frame templates, along with instructions for making and using them, are available at <http://www.cogain.org/faq/eye-gaze-communication-board> (accessed 1 March 2009).

directly to mouse movements to create an “eye mouse” may seem an easy solution; however, there are several issues that have to be taken into account.

Eyes move constantly, and they make small corrective movements even when fixating. If the cursor of an eye mouse were to follow eye movements faithfully without any smoothing, the cursor movement would appear very jerky and it would be difficult to concentrate on pointing, since the cursor itself would attract attention (Jacob, 1993). Applying proper smoothing (by averaging data from several gaze points) “dampens” the jitter, making visual feedback more comfortable and less disturbing (Lankford, 2000). Smoothing the cursor may also assist in keeping the pointer on the target long enough for it to be selected. On the other hand, smoothing slows down the cursor movement. Some applications, such as action games or the Dasher text entry system (Ward & MacKay, 2002), benefit from faster response. Thus, it should be possible to adjust the amount of smoothing (Donegan et al., 2006a).

If calibration is poor, the cursor may not be located exactly where the user looks, but a few pixels offset. As the user tries to look at the cursor, which is displaced from the actual point of gaze, the cursor again moves away from the gaze point when the gaze is moved to look at the cursor. This causes users to chase a cursor that is always a few pixels away from the point they are looking at (Jacob, 1995). Experienced users may learn either to ignore the cursor or to take advantage of the visual feedback provided by the cursor in order to compensate for any slight calibration errors by adjusting their gaze point accordingly to bring the cursor onto an object (Donegan et al., 2006b).

If screen resolution is set low and large icons are used, people with good, stable eye control may be able to use standard graphical user interfaces (such as Windows) directly by eye gaze (see, e.g., Donegan et al., 2006b). Special techniques, such as zooming or temporarily enlarging an object on the screen (Bates & Istance, 2002; Skovsgaard et al., 2008), or a fisheye lens (Ashmore et al., 2005), aid in selecting tiny objects such as menu items or shortcut buttons in a typical Windows environment. Figure 4.2 shows an example screenshot of use of the Zoom tool included in Quick Glance’s (Rasmusson et al., 1999) Eye Tools menu to magnify a portion of an image.

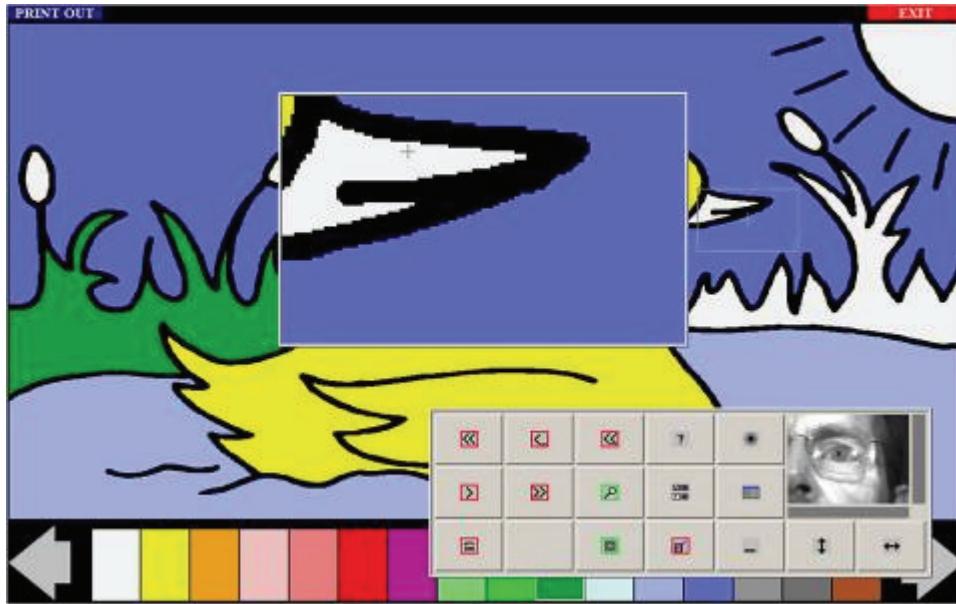


Figure 4.2. Quick Glance's Eye Tools (EyeTech, 2005) menu provides a zoom tool for target magnification, in addition to mouse actions (double-click, right click, dragging, etc), quick calibration correction, and other options.

A mouse click can be executed by dwell time, blink, switch, or any other selection method described previously. In addition to single (left) mouse click, the actions of right click, double-click, and dragging are needed if full mouse emulation is desired. These functions are typically provided in a separate ("mouse click") menu, such as Quick Glance's Eye Tools menu shown in Figure 4.2. An alternative solution is, for example, to use a short dwell time for a single click and longer dwell time for a double click (Lankford, 2000). Feedback on the different stages of dwell time progress can be shown via the cursor itself by changing its appearance. It may, however, be difficult for some people to understand the different stages.

The main benefit of using mouse emulation is that it potentially enables access to *any* graphical interface based on windows, icons, menus, and pointer devices (WIMP). In addition, and as importantly, it enables the use of any existing accessibility software, such as environmental control applications or "dwell click" tools.

For example, there are a number of virtual keyboards aimed at users unable to use a standard keyboard or mouse.⁷ The on-screen keyboards can typically be operated by a conventional mouse, or by an alternative input device that can trigger mouse events. Therefore, any eye tracking

⁷ Examples include WiViK[®] (see <http://www.wivik.com/>), SoftType (see <http://www.orin.com/>), and many more.

system that can emulate the mouse can probably be used to control such keyboards. There may be advantages in using existing on-screen keyboards, such as avoiding the need to redo well-done work such as implementing a word prediction system and icons that activate various window handling commands. Many of these have already been tested with disabled users and have advanced setup features for customizing the keyboard to meet the user's individual needs. They emulate the standard keyboard and can be used to control several (if not all) standard applications. For communication purposes, many of them include support for synthesized speech output. Many of the on-screen keyboard programs can also display graphics, pictures, and symbols for those who have difficulties with text (Nisbet & Poon, 1998).

Many of the on-screen keyboards that are specially made for disabled users include a set of on-screen keys to emulate mouse control and other special features like "sticky keys" for selecting key combinations such as Ctrl+C, support for scanning methods, word prediction, predefined phrases, and support for environment control. Many of them are fully integrated into the operating environment, meaning that they can be used to control most of the standard applications, like WordPad in Microsoft Windows.

There are also applications that allow the user (or the user's helper) to design special access keyboards need-specifically, with varying target sizes and layouts (for more information on the possibilities and for example applications, see Donegan et al., 2005 or visit the COGAIN Web site⁸).

Eye mouse versus head mouse

Perhaps the closest alternative to eye pointing is using the head to point, if the user retains some head control. Obviously, the main difference between eye pointing and head pointing is that, when one is pointing with a head mouse, the eyes are free for viewing. If the user has good head control, a head mouse⁹ can also be quite accurate. Given the availability and price of head mice (even a "mid-price-range" eye control system would be far more expensive than a head mouse), a head mouse could be a better choice than an eye mouse for those who can use it. It should be

⁸ Examples of individually made, adjustable keyboards are available for download at no charge via http://www.cogain.org/user_involvement/exemplars (accessed 1 March 2009).

⁹ For more information about head pointing and examples of such "head mice," see, for example, <http://abilitynet.wetpaint.com/page/Head+Tracking> (accessed 1 March 2009).

noted, though, that anecdotal evidence suggests concerns over prolonged exposure of the neck to repetitive pointing tasks.

Bates and Istance (2002, 2003) compared the eye mouse and head mouse in a “real-world” test that consisted of various simple tasks using a word processor and an Internet browser. Overall, performance and user satisfaction were higher with the head mouse than for the eye mouse. However, the results suggest that an eye mouse could exceed the performance of a head mouse and approach that of a hand mouse if the target sizes were large. Performance increased with practice. Experienced eye mouse users reached head mouse performance levels (though it seems to require more training to master an eye mouse than a head mouse).

Hansen et al. (2004) obtained similar results when comparing eye typing with input by head or hand. They tested eye performance with the on-screen keyboards Dasher (Ward & MacKay, 2002) and GazeTalk (Hansen et al., 2001) in Danish and in Japanese. Gaze interaction was found to be just as fast as head interaction but more erroneous than use of a head or hand mouse.

Optionally, the user could have a choice between an eye and head mouse, depending on the task and the physical condition of the user, as was suggested by a user who tried eye control and was impressed by it (Donegan et al., 2005). For her, eye control felt more natural and required less effort than either the mouthstick (her main interaction method) or head mouse.

4.2 ASSISTIVE APPLICATIONS OF EYE TRACKING

Typing with the Eye and Communication

Communication is a fundamental human need, and difficulties in communication may lead to loneliness and social isolation. Developers of eye control systems are well aware of this, so eye typing is typically the first application implemented and tried out by users with an eye control system. Eye typing systems have been available since the late 1970s (Majaranta & Riih , 2002). Since text entry by eye gaze is the topic of this thesis, only a brief, general introduction is given here.

In a typical eye typing system, there is an on-screen keyboard (there is no need to adhere to a QWERTY layout). The user types by looking at the characters on the keyboard and selects them via dwell time or by using any of the selection methods discussed earlier. Typed text appears in the input field, often located above the keyboard (Figure 4.3). Different eye typing methods are further discussed in Chapter 5.

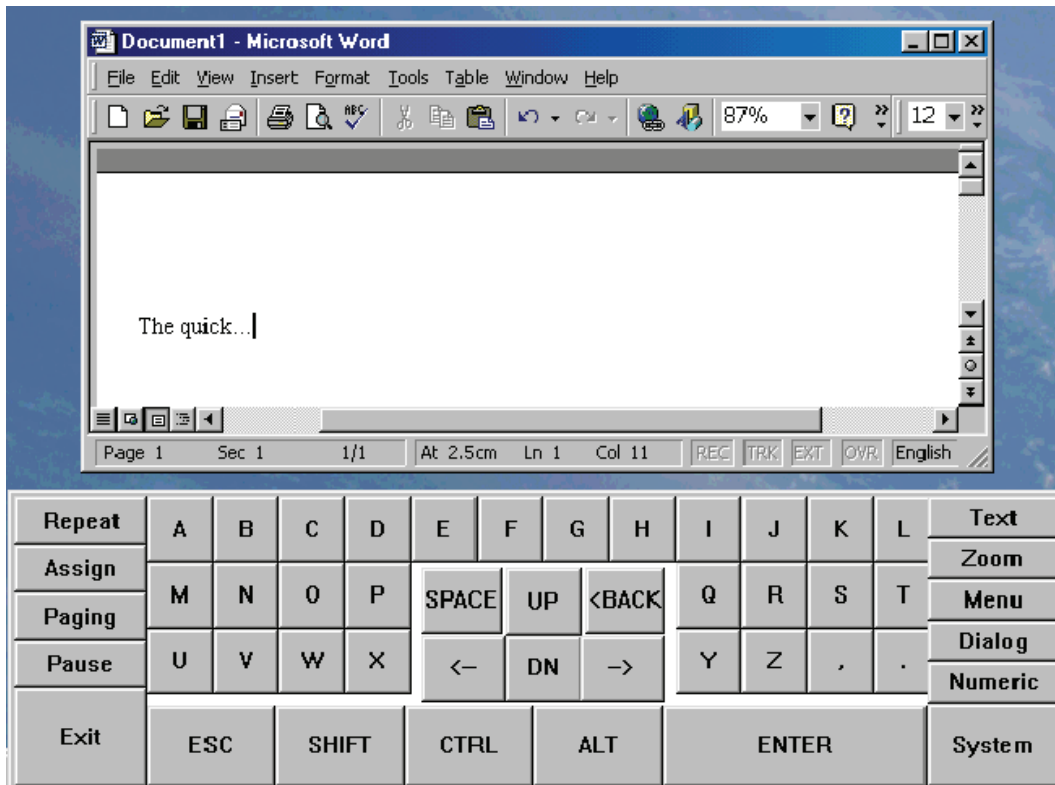


Figure 4.3. EC Key (Istance et al. 1996), a typical gaze-driven keyboard (image courtesy of Richard Bates).

Communication is more than just production of text, and eye typing feels especially slow in face-to-face communication situations, since its entry rate (only a few words per minute) is far below that of human speech (150–250 wpm). The goal of the user is not just to type letters but to produce phrases and sentences. Writing a message letter by letter with the eye is relatively slow. Phrases for everyday usage could and should be included in the program. The system should support editable phrases, because the needs of disabled users vary greatly. Since not all of the phrases can be visible at a time, they can be arranged into a tree structure, as in the phrase selection menu of LC Eyegaze (Chapman, 1991) (see Figure 4.4).

The gaze communication system may also have a sentence buffer for predefined strings that can be joined together. The Eyetracker communication system developed by Friedman et al. (1982) provides the option to first select a standard phrase (e.g., “please give me”) and then complete it with another (“a drink of water”). Ready-made greetings and phrases can speed up everyday communication, and a speech synthesizer can give the user a voice and the ability to speak aloud. However, the synthesized voice may not feel “right” if it does not match the user’s age and gender (Friedman et al., 1982).

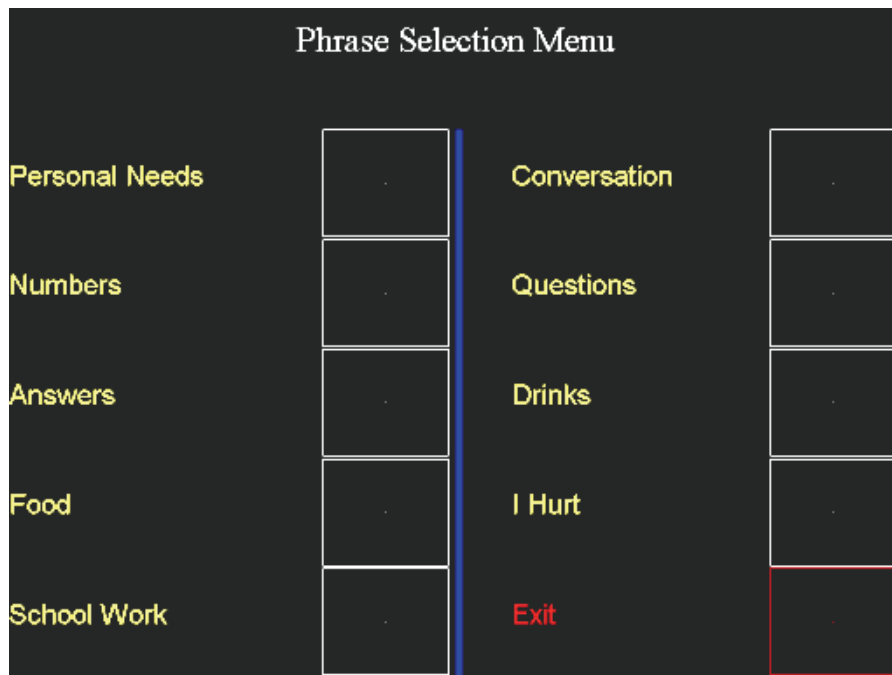


Figure 4.4. The phrase selection menu of LC Eyegaze (Chapman, 1991)
 (image courtesy of Nancy Cleveland, LC Technologies, Inc., <http://www.eyegaze.com/>).

Not all disabled people are able to read or write; instead, some use pictures and icons for communication. Therefore, it is useful for an eye-gaze-based communication system to support pictures in addition to text. There are several kinds of communicative pictures in use, including PCS (Picture Communication Symbols), Rebus, Makaton, Minspeak, Picsyms, and Bliss (MacDonald, 1998). To meet the various needs of disabled users, the choice of symbol set should also be customizable.

That the user must look at a computer monitor in order to communicate greatly alters the communication between the user and other people. The normal eye-to-eye contact of typical communication is broken, and facial expressions cannot be so easily viewed by a user whose attention is focused on the screen. Because of this loss of facial communication, a see-through communication board (Goossens' & Crain, 1987; Scott, 1998) may feel more natural for everyday communication, since the people involved maintain a face-to-face connection (see Figure 4.1). The communication board can also be used everywhere, and it is always reliable and does not “crash.” Besides, an experienced human interpreter¹⁰ is a far more effective word predictor than any of the computer-based systems, which

¹⁰ After years of practice, one may learn the locations of letters and thus not need the board anymore; each letter has its position in thin air (see http://www.cogain.org/media/visiting_kati, accessed 1 March 2009).

do not understand conversational context and situation and are not able to understand humor, etc. There is still a need for more effective textual gaze communication.

Drawing with the Eye

It is simple to bind mouse movement to eye movement and then just select the paint tool, as is shown in Figure 2.2, where a thick line with varying colors follows the user's eye movements. However, drawing recognizable objects (houses, trees, or people) is not easy with "free-eye drawing" (Tchalenko, 2001). The characteristics of eye movements prevent using the eyes as a pencil to sketch a finely defined shape. For example, since eye movements are ballistic, it is easy to draw a fairly straight direct line from point A to B just by glancing at the starting and ending point. However, trying to draw slowly or trying to draw a curved line is hard, since the eye does not easily fixate on empty white space and moves not smoothly but in saccadic jumps. Thus, an eye-drawn circle would not be a smoothly curved circle; it would be more like a multi-angled polygon with many small jumps (Heikkilä, 2008). The human eye needs a moving target to follow in order to initiate smooth (pursuit) movement.

The Midas touch problem is also strongly present: is the user moving the eyes to draw, or just looking at the drawing? A method for distinguishing between "drawing" and "looking" is needed (Yeo & Chiu, 2006). Hornof et al. (2004) developed EyeDraw (see Figure 4.5), which implements a set of eye-controlled tools for drawing and painting. Using the tools, the user manages the drawing process with assistance, rather than attempting free-eye drawing. For example, in order to draw a line, the user first looks at the "draw line" button in the tool menu. The button is highlighted after dwelling on it, to show that the "draw line" tool is active. To draw the line, the user then moves the cursor to the location where the drawing should start and dwells on it. The cursor color changes to show that the starting point has been defined. From now on, a straight line, with its other end fixed on the starting point, starts to follow the user's gaze. Again, the user has to dwell on the selected location to define an ending point for the line. By changing the color of the cursor, the program provides feedback on the current state, which can be either "looking" or "drawing."

Defining the starting point by staring at a blank drawing surface is somewhat difficult, as the eye does not easily fixate without a target object of interest. Therefore, EyeDraw provides a grid of dots that act as visual anchors and aid in placing the starting and ending points, literally, on the dot. EyeDraw has been successfully used by people with disabilities to produce drawings, although younger children may get frustrated since they may not have the patience to learn the tools and different states (Hornof & Cavender, 2005). For them, free-eye drawing provides an easy start with immediate positive feedback.

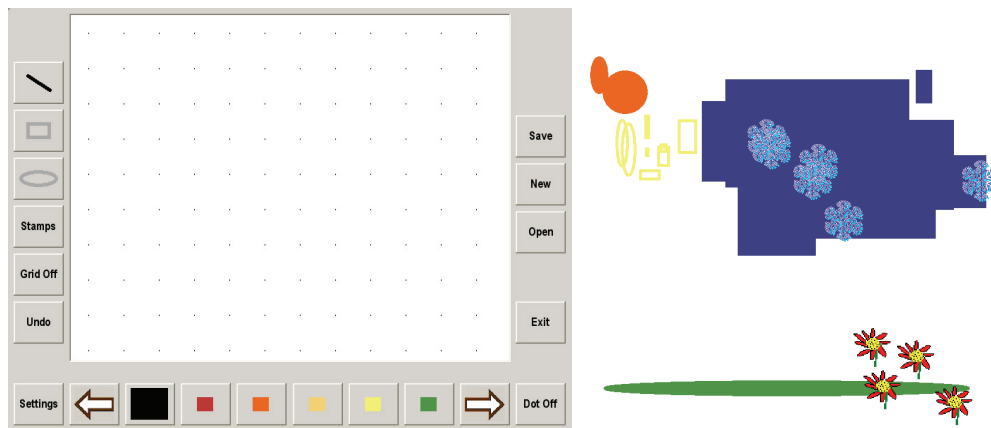


Figure 4.5. EyeDraw¹¹ (Hornof et al., 2004) and a landscape (on the right) drawn by a young woman using EyeDraw (image courtesy of Dr. Anthony Hornof).

Beyond Communication and Control

In addition to eye typing and eye drawing, there are several dedicated eye-controlled applications, such as e-mail, Internet browsing (Castellina & Corno, 2007; Nakano et al., 2004), accessing online libraries (Lund & Hansen, 2008), games (Dorr et al., 2007; Isokoski & Martin, 2006; Isokoski et al., 2009; Smith & Graham, 2006), and interaction with online virtual communities (Bates et al., 2008; Vickers et al., 2008). Some of the applications, such as games and Internet browsing, are included in many of the commercial eye control systems targeted at people with disabilities (see Table 2.1).

The main advantage of having “dedicated” applications developed especially for eye control (instead of using standard applications via mouse emulation) is that the many special requirements of gaze interaction (Donegan et al., 2009) can better be taken into account. For example, the layout and the structure of the application can be designed such that items are large enough to be easily accessible by gaze. Similarly, the feedback provided by a dedicated application can be implemented to support the process of gaze pointing and dwell time selection.

A current trend in gaze-controlled applications seems to be to move gaze interaction away from the desktop environment, to support environmental control by gaze (Bonino et al., 2009; Corno et al., 2009) and gaze-based mobility (Barea et al., 2002; Novák et al., 2008). Another exciting area of development is support for gaze control of physical objects and home

¹¹ EyeDraw is available for download at <http://www.cs.uoregon.edu/research/cm-hci/EyeDraw/> (accessed 1 March 2009).

appliances (Shell et al., 2003). One example of such gaze interaction with physical objects is the gaze-controlled toy car developed at Czech Technical University (Fejtová et al., 2006), illustrated in Figure 4.6.

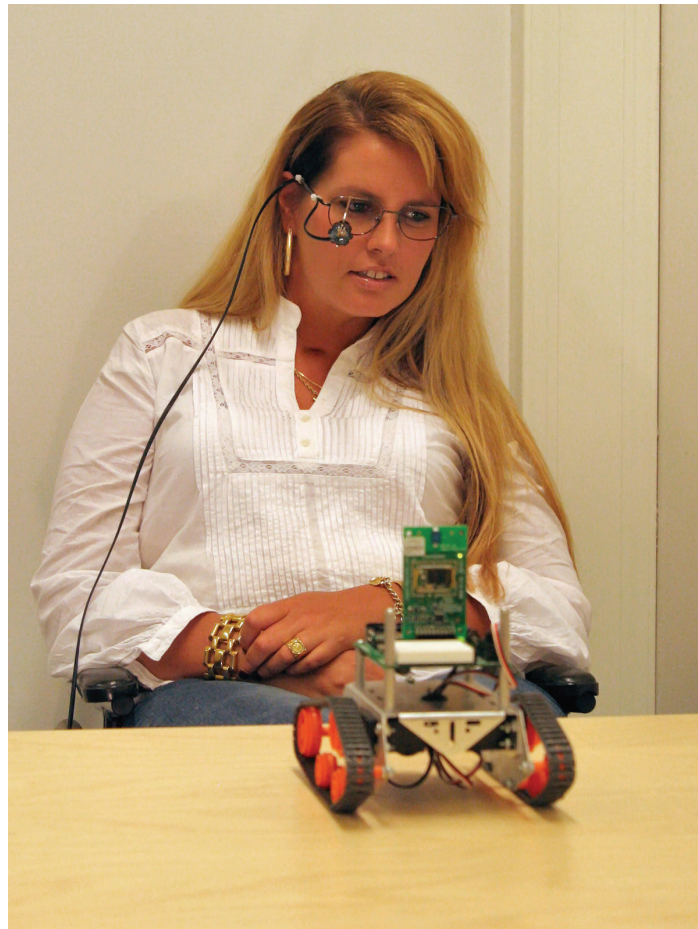


Figure 4.6. The woman in the picture is remotely controlling a toy car with her eyes. The I4Control¹² eye tracker acts as a gaze-controlled joystick. By looking up, she sends a “forward” command to the toy car and by looking left she can make the car turn “left” (photo: Niina Majaranta).

An extensive study of user requirements by Donegan et al. (2005) shows that, thus far, eye control can effectively meet only a limited range of user requirements, and that it can be used effectively by only a limited number of people with disabilities. Furthermore, the range of applications that are suitable for easy and effortless control with the eye is limited.

In lists of potential user groups of eye control technology, people with ALS are usually among those with the highest priority, as people who most need and benefit from eye control (Calvo et al., 2008). In the late

¹² For more information about I4Control, see <http://www.i4control.eu/> (accessed 1 March 2009).

stages of ALS, control over all other body movements may be lost, but the person can still move the eyes. For these people, eye control is a *necessity*.

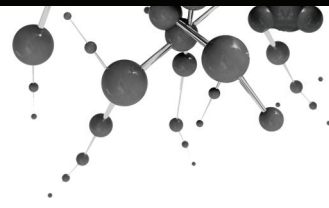
Eye control suits people with ALS well because they have good visual, cognitive, and literacy skills. They also do not have involuntary movement, so their eyes are fairly easy to track. However, there are people with a wide range of complex disabilities who might also benefit greatly from eye control but who find it difficult because of involuntary head movement, visual difficulties, or learning difficulties. Recent advances in technology have considerably improved the quality of eye tracking systems, such that a far broader group of people may now benefit from eye control. Eye control can be a genuine *choice* for both types of users, since eye control can be faster and less tiring than, for example, a head-pointing-based system or a manual-switch-based system (Garbe, 2006; Donegan et al., 2009).

Even if the first trial with eye control fails, this does not necessarily mean that eye control technology is not suitable for the user concerned. With appropriate (or properly adjusted) hardware and software, eye control technology can become accessible even to people with the most complex of disabilities (Donegan & Oosthuizen, 2006).

In Europe alone, the number of potential beneficiaries of eye tracking technology amounts to several hundred thousand people, but, as yet, only a small number of these people are actually using eye control (Jordansen et al., 2005). For many of them, eye control is potentially the quickest, least tiring, and most reliable form of access to technology – by far.

Many of the potential users are already using special software for communication. If their condition deteriorates or if they for any other reason move to eye control, they should be able to continue using familiar programs. Existing applications should be made as “eye-friendly” as possible (Donegan et al., 2006a). There are a variety of applications directed especially at users with disabilities, such as environmental control applications, that would be highly beneficial for eye control users if they were made eye-friendly.

Eye control can offer great possibilities, but it is also important to understand its limitations. As a mother of a young boy noted after an eye control trial, “it was ironic that the more fun he had, the more he laughed and the more his eyes closed, making it impossible for the tracker to pick up his eyes.”



5 Text Entry by Gaze

Systems that utilize eye tracking for text entry have existed since the 1970s (Majaranta & Riih , 2002). In fact, the first real-time applications to use eye tracking in human–computer interaction were targeted at people with disabilities (Jacob & Karn, 2003).

The eyes can be used for text entry in various ways. We (Majaranta & Riih , 2007) have categorized text entry methods according to input technique. We start with text entry by direct gaze pointing, “gaze typing” (or “eye typing”), wherein the user enters text by “pressing” keys on a virtual keyboard one at a time (addressed in Section 5.1). Text entry by eye switches enables gaze-based text entry for people who have only limited eye movements (described in Section 5.2). Text entry by discrete, consecutive gaze gestures is introduced in Section 5.3. Finally, text entry via continuous gaze gestures, “gaze writing” (note the difference from “gaze typing”), is introduced in Section 5.4. A similar categorization is used by Bee and Andr  (2008), who distinguished among three types of writing: typing, gesturing, and continuous writing.

5.1 TEXT ENTRY BY DIRECT GAZE POINTING

The most common way to use gaze for text entry is direct pointing by looking at the desired letter. A typical setup has an on-screen keyboard with a static layout, an eye tracking device that tracks the user’s gaze, and a computer that analyzes the user’s gaze behavior (see Figure 5.1).



Figure 5.1. The Tobii eye tracking device has a camera integrated into the frame of the monitor that shows the on-screen keyboard (photo: Henna Heikkilä).

To type by gaze, the user *focuses* on the desired letter by looking at one of the keys on the on-screen keyboard. The system gives *feedback* on the item in focus, for example, by highlighting the item or by moving a “gaze cursor” over the key in focus. Once it has the focus, the item can be *selected* via, for instance, a separate switch, a blink, a wink, or even a wrinkle or any other (facial) muscle activity. The typed letter appears in the text field, often located above the keyboard. The system may also give feedback on successful selection by speaking out the letter or playing a “click” sound (different feedback methods are reviewed and further discussed in Chapter 8).

For severely disabled people, *dwell time* is often the best and the only means of selection. As discussed earlier, dwell means a prolonged gaze: the user needs to fixate on the key for longer than a predefined threshold time (typically, 500–1000 ms) in order for the key to be selected. If dwelling is used for focusing, the system usually provides the user with an indication of the progression of dwell time (see Chapter 8). Dwell-time-based eye typing can be slow, typically below 10 wpm, because dwell time durations set a limit on the maximum typing speed. For example, with a 500 ms dwell time and a 40 ms saccade from one key to another, the maximum speed would be 22 wpm. In practice, entry rates are far below that (Majaranta & Rähkä, 2007). People need time for cognitive processing, to think what to type next, to search for the next key on the keyboard, to correct errors, etc. Experienced users may require considerably shorter dwell times (as low as 200–300 ms), which, naturally, increases the text

entry rate correspondingly (to be as high as 20 wpm); see Chapter 9 for more information about learning gaze typing and adjusting dwell times.

Typically, with a “flat” keyboard layout design where all characters are visible, only one keystroke per character (KSPC) is needed since most letters can be directly pointed at and selected. As discussed earlier, sometimes the full keyboard cannot be shown at once, on account of accuracy and calibration issues. In such a case, only a few keys can be shown at a time, which prevents the use of full-size keyboards such as a full QWERTY keyboard. Similarly to the use of several key presses to enter a letter in the multi-tap method for mobile phones, large on-screen keys can be selected repeatedly to enter a character, which increases the KSPC figure.

Via on-screen buttons (instead of physical buttons), keys and controls can be organized hierarchically in menus and sub-menus, and special techniques such as automatic word prediction can be used to speed up the text entry process, with constantly changing and adapting keyboard layouts (see, for example, Frey et al., 1990; Hansen et al., 2003b).



Figure 5.2. GazeTalk¹³ (Hansen et al., 2001) provides big buttons that are easy to hit. To speed up typing, it provides a list of the next probable words, predicted on the basis of the text written so far. If the user selects the cell with the list of words, the cells that now contain individual letters will be filled in with the words.

¹³ GazeTalk is freely available at <http://www.cogain.org/downloads> (accessed 1 March 2009).

As an example, GazeTalk (illustrated in Figure 5.2) has large buttons that support users who have difficulty obtaining or maintaining good calibration, or it may be used to enable use of an eye tracker with a low spatial resolution. It also aids with typing speed by changing the keyboard layout, using a language model to predict the most probable next letters (see Section 6.3 for more information about character prediction). There are also other reasons for reducing the number of keys, such as to save screen space, as discussed in Chapter 7.

5.2 TEXT ENTRY THROUGH EYE SWITCHES

Some people may have difficulties in fixating because of their physical condition or state of health (Donegan et al., 2005). They cannot keep their gaze still for the time needed to focus. The user may also be able to move his or her eyes in one direction only (e.g., in locked-in syndrome – see Chapman, 1991). In such cases, other methods for selecting an item by gaze are needed.

Voluntary eye blinks or winks can be used as binary switches (see, e.g., Grauman et al., 2003). For text entry, blinks are usually combined with a scanning technique, with letters organized into a matrix. The system moves the focus automatically by scanning the alphabet matrix line by line. The highlighted line is selected by an eye blink. Then the individual letters on the selected line are scanned through and again the user blinks when the desired letter is highlighted (see Figure 5.3).

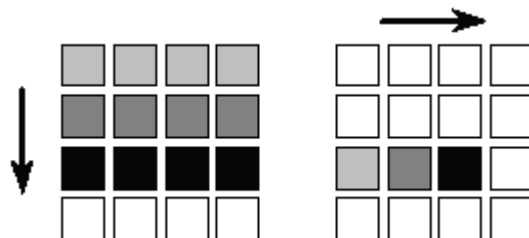


Figure 5.3. Sequential row-column scanning (Shein, 1997).

Text entry with scanning is obviously slow, since the user needs to wait for the scan routine to go through all the rows and columns until the desired item is reached (around 2–6 wpm, Beukelman & Mirenda, 1992; Koester & Levine, 1994a). In addition, there is a brief delay (similar to dwell time) between scanned items, giving time for the user to react and select it; the duration of this delay naturally limits the maximum entry speed. The process can be sped up by optimizing the scan interval (Evreinov & Raisamo, 2004), or scanning a list of (predicted) words instead of single characters and organizing the items into groups according to probabilities or semantic coding (Demasco & McCoy, 1992). In addition to the automatic scanning method described here, there are

also a number of variants and more advanced scanning methods available (Shein, 1997), beyond the scope of this thesis.

In addition to eye blinks, coarse eye movements can be used as switches. The *Eye-Switch Controlled Communication Aids* of ten Kate et al. (1979) used large (about 15-degree) eye movements to the left and to the right as switches. This system, too, applied the scanning technique. The user could start the scanning by glancing to the left and select the item currently with the focus by glancing to the right.

I4Control (Fejtová et al., 2004), illustrated in Figure 4.6, can be considered to be a four-direction eye-operated joystick, since looking in any of the four directions (left, right, up, down) causes the cursor to move in that direction until the eye returns to the home position (center). The cursor can also be stopped by blinking. A blink emulates a click or double click. Text entry in *I4Control* is achieved by moving the mouse cursor over the desired key on an on-screen keyboard and selecting the key with a blink. The four different eye movements recognized by *I4Control* can be considered four switches, each activated by a different glance or gaze gesture.

5.3 TEXT ENTRY BY DISCRETE GAZE GESTURES

Several discrete gaze gestures can also be combined into one operation. This approach is used in *VisionKey* (Kahn et al., 1999), where an eye tracker and a keyboard display (Figure 5.4, right) are attached to eyeglass frames (on the left in Figure 5.4).

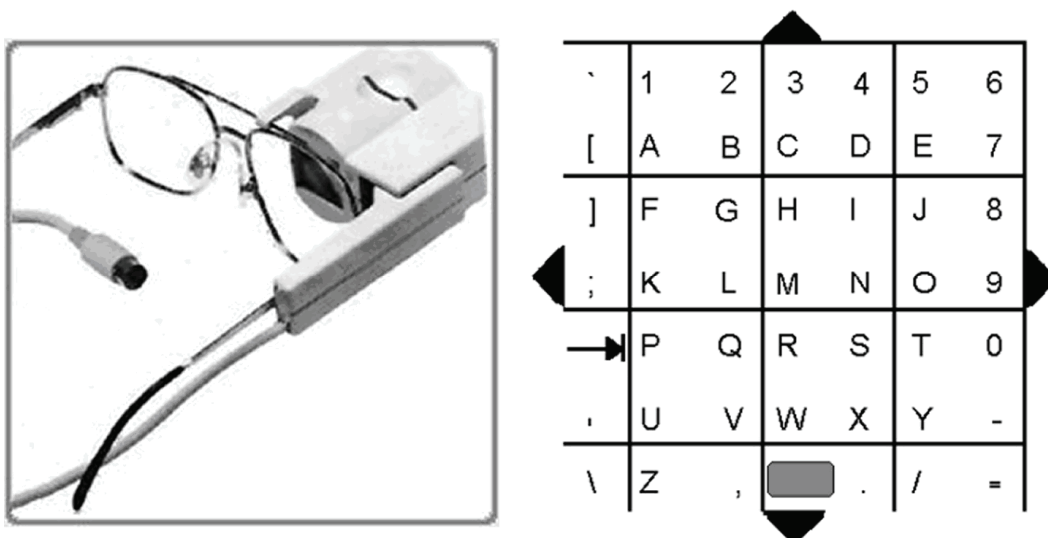


Figure 5.4. VisionKey is attached to eyeglass frames (left). It has a small screen integrated into it (right) (from H.K. EyeCan Ltd., <http://www.eyecan.ca/>, reprinted with permission).

Because the key chart is attached in front of the user's eye, it is important to make sure that simply staring at a letter does not select it. The VisionKey selection method avoids the Midas touch problem by using a two-level selection method - i.e., two consecutive gaze gestures for activating a selection. To select a character, the user must first gaze at the edge of the chart that corresponds to the location of the character in its block. For example, a user wanting to select the letter "G" first glances at the upper right corner of the chart and then looks at the block where "G" is located (or simply looks at "G"). After a predefined dwell time, the selected key is highlighted to confirm a successful selection.

Isokoski (2000) discussed the potential of gaze-based text entry with several (single) switch- and gesture-based systems such as Morse code, Quikwriting (Perlin, 1998), Circular Cirrin (Mankoff & Abowd, 1998), and Minimal Device Independent Text Input Method (MDITIM) (Isokoski & Raisamo, 2000). All of these have potential to be adapted for gaze-based text entry with off-screen targets, which would save the precious screen space. Isokoski experimented with one of these systems, *MDITIM*. It enables text entry with four strokes (or button clicks): north, east, south, and west. Isokoski placed paper targets on each side of the monitor, and the user could enter a character by looking at each of the targets, in a specified order. In addition, he had one extra target, a "modifier," in the NW corner of the monitor, for uppercase and other secondary interpretations of the characters. Isokoski's paper has prompted several further studies (as acknowledged by the authors), some of which are summarized below.

Porta and Turina (2008) developed *Eye-S* to take advantage of gaze gestures for both text entry and control of computer applications. Instead of off-screen targets, they preferred nine on-screen areas that act as hot spots for the start and end points of the eye strokes. The user enters a character or command by glancing at the hot spots in the order specified for the desired gesture. Casual viewing (the natural gaze path) on the screen does not launch a gestural command. Usually the hot spots are invisible. However, in the learning mode they were shown as transparent squares (see Figure 5.5). The advantage of using on-screen targets is that the system can give active feedback during the process. The hot spot is highlighted after a brief dwell time (400 ms) and the number "1" is shown to indicate that the gesture recognition process has started. If the user looks at another hot spot within a set time (1000 ms), it is highlighted and "2" is shown. When the third or fourth stroke has landed, the target is again highlighted, and if the gesture's end point has been reached, the recognized character (or command) is shown. The gesture alphabet of *Eye-S* is similar to the glyphs or "graffiti" used in personal organizers, and the "eye graffiti" suggested also by Milekic (2003). In both, the gestures resemble strokes for drawing a letter by hand. Experiments with two

experienced users show that Eye-S could be used successfully to enter text with a speed of 6.8 wpm.

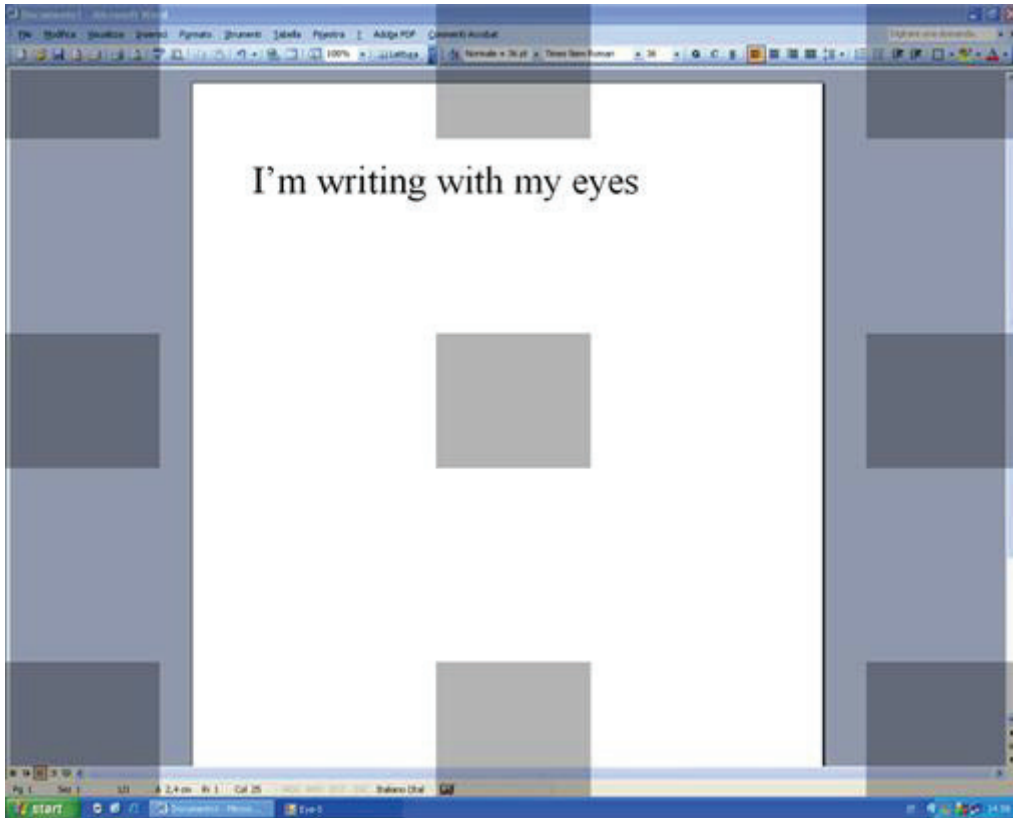


Figure 5.5. Eye-S (Porta & Turina, 2008) with hot spots shown (image courtesy of Dr. Marco Porta, Universita' di Pavia).

Wobbrock et al. (2008) adapted EdgeWrite (Wobbrock et al., 2003) for gaze control and called it *EyeWrite* (illustrated in Figure 5.6). In both, the user enters characters by drawing letter like gestures in a small dedicated window. The advantage of a separate input window is that there is no danger of the stroke overlapping with other user interface objects, which ensures robust interpretation of the gestures. The window also enables dynamic feedback while the user enters the strokes needed for each gesture. As soon as the gaze enters the active area, an arch is drawn there to illustrate a successful stroke and the target character recognized is shown in the final target area.

The input window has visually separated corners that act as target areas for the strokes. The user draws the letters by moving the gaze (cursor) from one corner to another. For example, the letter "t" is entered by moving the cursor (gaze) from the top left corner to the top right corner, then to the bottom right corner (see Figure 5.6). The interpreted character is sent to a separate text input window, which has the focus (e.g., Notepad). No dwelling inside the corner area is needed; the stroke is recognized as soon as the cursor crosses the line defining the corner area. The system is very resistant to noise in input (inaccuracy in eye tracking),

and crossing is also generally easier than pointing for large, approximate targets, according to Fitts's law (Wobbrock & Myers, 2006).

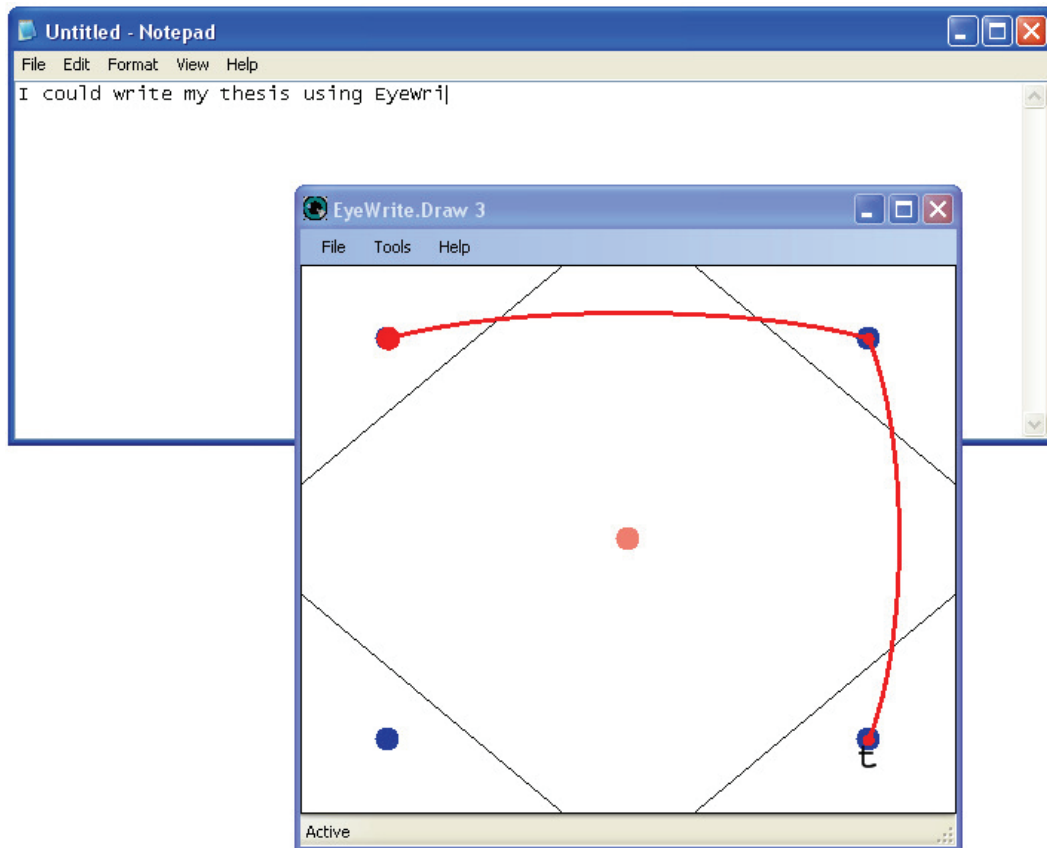


Figure 5.6. EyeWrite (Wobbrock et al., 2008) in action, with the letter “t” being entered.

Wobbrock et al. (2008) conducted a longitudinal experiment to compare the gaze-gesture-based EyeWrite with an on-screen keyboard based on direct gaze pointing and dwell time. The average speed was 4.87 wpm for EyeWrite and 7.03 wpm for the on-screen keyboard. There were fewer errors left in text with EyeWrite, but there was no significant difference in the number of errors corrected during entry. The on-screen keyboard was shrunk such that its size approximately matched that of EyeWrite (400 x 400 pixels), which probably affected both speed and accuracy – small targets are hard to hit – making it difficult to interpret these results.

In the study, VisionKey, Eye-S, and EyeWrite all required a brief dwell time for segmentation. VisionKey required a brief dwell that confirmed the selection, in Eye-S the brief delay on a hot spot was used to initiate a gesture, and in EyeWrite the brief dwelling on the center helped to differentiate gestures from each other. Huckauf and Urbina (2007; see also Urbina & Huckauf, 2007) developed several methods that do not require any dwell time in any phase of the writing process, as described below.

Huckauf and Urbina (2007) developed a dwell-time-free text entry system that takes advantage of pie menus, called *pEYEs*, or *pEYEdit* or *pEYWrite* (see Figure 5.7). Since bigger sectors are easier to select, letters are grouped

into the sectors of the pie. To enter a letter, the user moves the cursor (by gaze) such that it crosses the outer part of the sector that contains the desired letter. A sub-menu with a separate sector for each of the letters opens immediately without the need for dwelling on it. The target letter is selected by glancing at (or over) the outer part of the sector where the desired letter is located. Again, no dwell time is needed. Since entering a letter requires the selection of two sectors, two strokes are needed for one character.

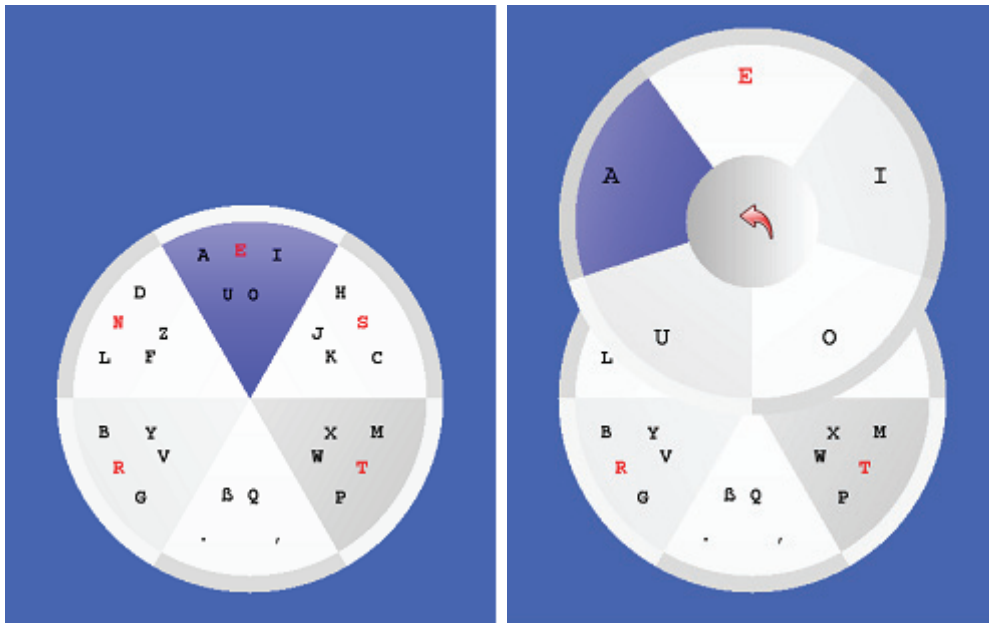


Figure 5.7. pEYWrite (Huckauf & Urbina, 2007) uses pie menus for text entry. In the figure, the letter “A” is being selected.

Having a separate selection area near the outer edge of each sector ensures that the user can look at the letters without them being selected. On the other hand, the user does not need to fall to and dwell on the selection area but can instead overshoot through it since the sector is selected as soon as the cursor (following the user’s gaze) crosses it. This makes the system less vulnerable to accuracy problems when compared to a system where the user needs to hit a small target area. Furthermore, the consecutive sectors can be selected immediately with strokes that follow each other, as if they were a single gesture. This is especially useful in cases where the target letter is in the same direction as the original sector, allowing the user to select both sectors in one large gesture towards the edge of the second menu. For example, in Figure 5.7 the letter “E” is located in the uppermost sector of each pie. Therefore, it can be selected with either two short glances up (selecting the inner sector first, then the outer sector) or in one long glance up (to overshoot the gaze gesture over both sectors). In an experiment, novice users could enter text at a speed of 7.85 wpm while an expert achieved 12.33 wpm (Huckauf & Urbina, 2008a).

Other dwell-time-free text entry systems developed by Urbina and Huckauf (2007) include *Iwrite* and *StarWrite*, which is a modification of *Iwrite*. In both, letters are arranged around the screen in alphabetical order. The user selects a letter by first looking at it and then “dragging” it to a special selection area on the bottom of the screen simply by looking at the selection area. Again, no dwell time is needed, since the user makes the gesture of glancing at first the letter, then the selection area. The selection is confirmed as soon as the gaze enters the area, enabling the user to continue typing without any pause between characters (or words, since the space character is among the characters entered in a similar way).

Urbina and Huckauf (2007) compared their dwell-time-free systems with Dasher (described in Section 5.4, below) and the standard QWERTY with 500 ms dwell time. They had disabled the language prediction features of Dasher for purposes of their experiment (since the other methods did not have any prediction features). The values reported are based on trials with two novices and one expert. The entry speeds achieved were

10.9 wpm (novices) and 15.8 wpm (expert) for QWERTY,

4.7 wpm (novices) and 7.4 wpm (expert) for Dasher,

6 wpm (novices) and 10.9 wpm (expert) for pEYEdit,

7.6 wpm (novices) and 11.4 wpm (expert) for *Iwrite*, and

5.9 wpm (novices) and 8.4 wpm (expert) for *StarWrite*.

Their comparison shows that direct gaze pointing with QWERTY was the fastest method for entering text by gaze, though there is some potential in the new approaches, to be investigated by Urbina and Huckauf in their future research.

Bee and André (2008) adapted *Quikwriting*¹⁴ (Perlin, 1998) for gaze input (as suggested by Isokoski in 2000 but never implemented by him). In the original *Quikwriting*, the characters are located in the active selection areas (sections). The characters are grouped such that the location of each indicates the gesture needed for entering it. Using a hand-operated input device, the user can first search for the target letter and then select it by moving the cursor (e.g., by mouse) to the selection area(s). The character is entered when the pointer is returned to the center. This is a problem with gaze input, since the user cannot visually search for the characters without immediately initiating the selection process. To prevent unintentional selection during visual search, Bee and André moved the characters into

¹⁴ *Quikwriting* is available for download at <http://mrl.nyu.edu/~perlin/demos/quikwriting.html> (accessed 1 March 2009).

the inner resting area, near the corresponding section (see Figure 5.8). The characters were still grouped such that the gesture needed is shown by the position of the character within the group. Furthermore, to support the user's selection process, each character from the group was shown in the adjacent sections as soon as the user initiated the selection by moving the cursor to one of the sections (away from the central area). Showing the characters within the sections eliminated the need to look at the central area if the user were to forget which section(s) he or she should select to enter the character. If the user had to look back at the center before finalizing the selection process, an error was likely to result, since a character is entered whenever the pointer is returned to the central area.

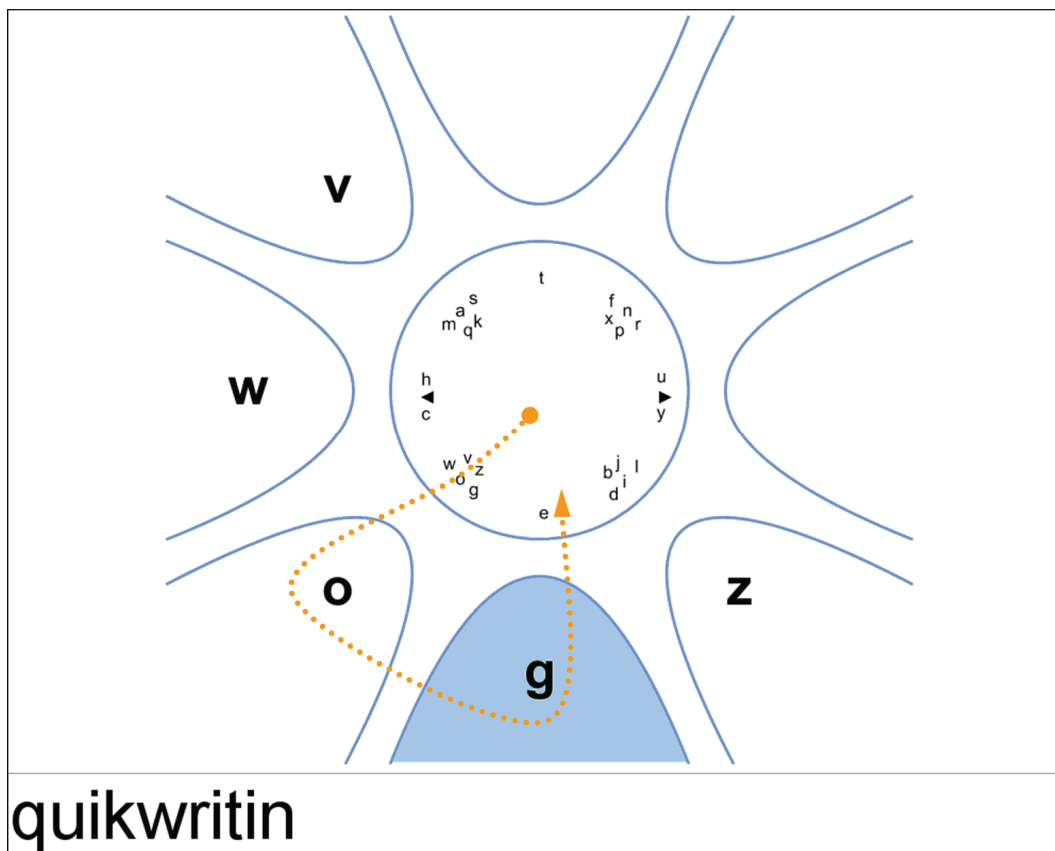


Figure 5.8. Quikwriting interface adapted for gaze. The figure illustrates the gaze path for gaze writing “g” (image courtesy of Dr. Nikolaus Bee, Augsburg University).

The location of the text input field where the characters entered appeared also caused problems in the gaze-based implementation: the user could not move the gaze from the central area to the text input area and back without crossing the active selection areas in between and thus unintentionally making selections. In their original implementation, Bee and André disabled the writing process when the user looked at the text input field and enabled writing when the gaze returned to the central area. However, after the user trial, they decided it was better to show the written text inside the central area, to minimize the swapping of the gaze

between the central area and the text input field (though only a small amount of the previously written text would fit there).

The adapted Quikwriting interface was tested against an on-screen keyboard, using a 750 ms dwell time. Participants achieved 5.0 wpm with Quikwriting and 7.8 wpm with the dwell-time-based keyboard. The result is encouraging, though it should be noted that it is based on only a very small sample, three novice participants (Bee & André, 2008).

Both Quikwriting and pEYEWrite enable continuous writing without a pause between characters (or words). The segmentation of the gestures is part of the process of moving the pointer (gaze) from one selection area to another. Nevertheless, the writing is still based on distinct gestures, even though some implementations enable entering them without a pause in between. In the following section, we will introduce text entry methods based on continuous pointing gestures. The user does not need to make any distinct gestures to enter characters; only one type of gesture is needed: pointing. However, here the pointing is different from the direct pointing used with the on-screen keyboards. The on-screen keyboard is static (even if implemented as sub-menus), and the user selects the letter by looking at the on-screen key. The text entry methods introduced below implement interfaces that change dynamically, and the user selects (groups of) characters by navigating through a world of characters that is continually changing. Thus, even if one can argue that this is indeed direct pointing, the pointing gesture is not static. It follows the dynamically changing interface, and the direction of the pointing changes smoothly and continuously while writing is taking place.

5.4 TEXT ENTRY BY CONTINUOUS POINTING GESTURES

Continuous writing can be especially useful for text entry by gaze because of the nature of human gaze. First, our eyes are always on (if not closed), so it can be compared to a pencil that is never lifted from the paper. Our eyes also move constantly; it is not natural for us to hold our gaze for long on a target. Even if we keep looking at one object, we usually make small saccades within (and around) the object.

Dasher (Ward & MacKay, 2002) is a zooming interface that is operated with continuous pointing gestures. In the beginning, the letters of the alphabet are located in a column on the right side of the screen (see Figure 5.9). Letters are ordered alphabetically (Figure 5.9, left). The user moves the cursor to point at the region that contains the desired letter, by looking at the letter. The area of the selected letter starts to zoom in (grow) and move left, closer to the center of the screen (Figure 5.9, right). Simultaneously, the language model of the system predicts the most probable next letters. The areas of those letters start to grow (as compared to other, less probable letters) within the chosen region. This brings the

most probable next letters closer to the current cursor position, thus minimizing distance and time to select the next letter(s). The letter is typed when it crosses the horizontal line at the center of the screen. Canceling the last letter is done by looking to the left; the letter then moves back to the right side of the screen.

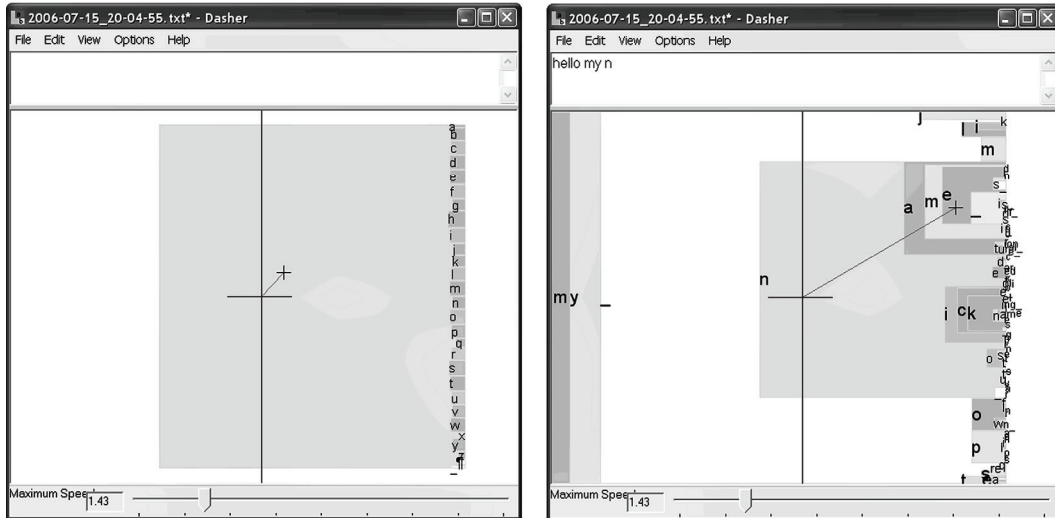


Figure 5.9. Dasher¹⁵ facilitates text entry via navigation through a zooming world of letters. In the initial state, the letters are ordered alphabetically on the right side of the screen (image on the left). As the user looks at the desired letter, its area starts to grow and simultaneously the language prediction system gives more space to the most probable next letters. In the image on the right, the user is in the middle of writing “name,” with “n” already selected.

Dasher’s mode-free continuous operation makes it especially suitable for gaze, since only one bit of information is required: the direction of gaze. No additional switches or dwell times are needed to make a selection or to cancel. Furthermore, instead of adding separate buttons or lists with the most probable next words, Dasher embeds the predictions in the writing process itself. Many successive characters can be selected via a single gesture, and often-used words are easier and faster to write than are rare words. This not only speeds up the text entry process but also makes it easier: In a comparative evaluation, users made fewer mistakes with Dasher than with a standard QWERTY keyboard.

The continuous gestures used in Dasher make it a radically different technique when compared to all of the others we have discussed. Some gestures can essentially select more than one character (the selection point does not have to be moved, since the display moves dynamically), which can speed up text entry. The most likely characters occupy a large portion

¹⁵ Dasher is freely available online at <http://www.inference.phy.cam.ac.uk/dasher/> and <http://www.cogain.org/downloads> (accessed 1 March 2009).

of the display space, so gestures do not always need to be accurate. Dasher can be used with any input device that is capable of gesturing. It has been implemented on a pocket PC to be used with a stylus and can even be controlled by breathing, via a special breath-mouse. A more detailed description and results from our experiment with Dasher can be found in Section 9.2.

Stargazer (Hansen et al., 2008) is another system that takes advantage of zooming. However, in *Stargazer*, the user zooms on the z-axis and panning occurs on the x- and y-axis. At the beginning, all characters are located in the space in a circular form in a familiar (in this case, alphabetical) order around the central area (see Figure 5.10, left). Special characters for backspace, undo, and stop actions are placed in the corners of the display (in some configurations, there is also an option for adjusting the speed, placed in one of the corners – see Hansen et al., 2008). The user navigates (“flies”) in the 3D space of characters by looking at the desired character. The 3D cursor (a “spaceship” made of three concentric circles) points at the direction of navigation. The display will pan towards the target character and that character will be moved to the center of the screen. The target character will start to grow bigger, indicating that the user is approaching it. Panning and zooming occur simultaneously in part, depending on the thresholds set for the central zoom area. Selection is performed by “flying” through the target letter. The user is always returned to the initial view after a selection is made.

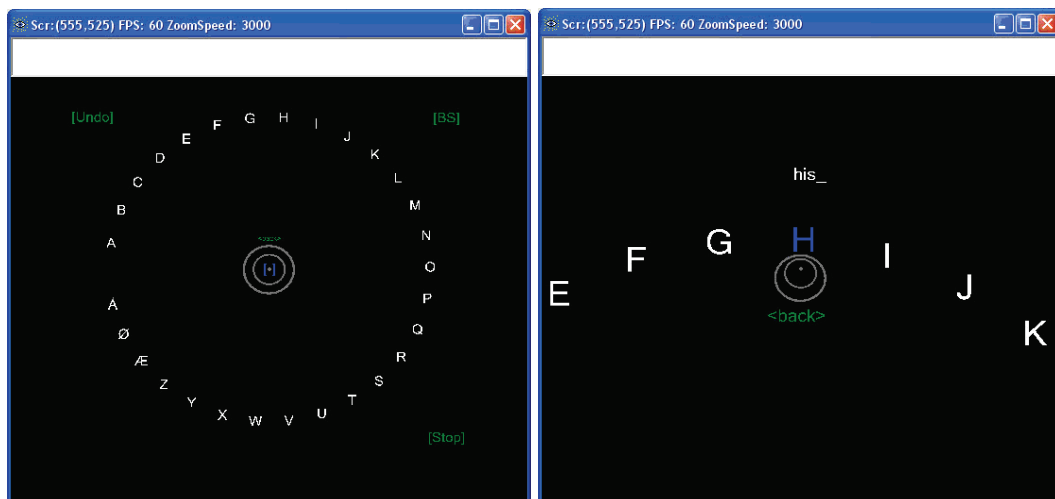


Figure 5.10. Writing with *Stargazer* (Hansen et al., 2008) always starts from the initial point in the center (illustrated on the left), from which the user navigates towards the desired character. The view pans and zooms toward the target (on the right).

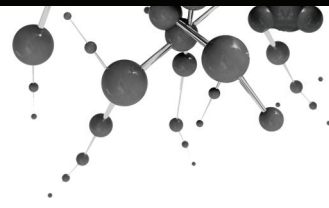
Zooming reduces the effects of noise. This enables use of low-resolution (low-accuracy) trackers and small display sizes. Another important advantage of a zooming interface is that a lot of information can be placed in a small screen space. In Figure 5.10, all characters fit in one circle. In some other configurations presented by Hansen et al. (2008), the characters are located in two concentric circles. Furthermore, as the

display zooms nearer to the target character, more information related to it can be shown. For example, in Figure 5.10, the most probable word has been predicted and shown above the focused character.

In an experiment with 48 participants, Hansen et al. (2008) found that novice users learned to write with Stargazer by gaze almost immediately. In the first try, they were able to write their name without losing the orientation. The grand mean of typing speeds was 3.47 wpm, which is fairly slow. However, in another experiment (with seven participants) it was found that after only five minutes of practice, participants achieved 8.16 wpm with a mean error rate of 1.23%, which is comparable with the speed achieved via a dwell-based on-screen keyboard.

Furthermore, Hansen et al. (2008) also introduced additional noise and latency to study how well the system tolerates these. The imposed noise slowed down typing speed because participants had to make additional corrections when the cursor was diverted from its course because of the noise. However, the participants were able to regain control and rarely lost the orientation. Similarly, adding latency slowed down typing but participants were able to cope with it, up to 200 ms. Even with 400 ms latency, writing was possible, though it was very slow, obviously.

Both Dasher and Stargazer initiate smooth pursuit eye movements while the user's gaze follows the target. Both are also mode-free and thus well suited to gaze input.



6 Character and Word Prediction

6.1 IN SEARCH OF BETTER TYPING SPEED

As discussed above (see Chapter 5), text entry by gaze is fairly slow, from only a few words per minute to around 10–15 wpm. Because the dwell time duration sets a maximum limit for the text entry rate, several attempts have been made to develop dwell-time-free systems (as introduced in the previous chapter). However, since dwell time was often replaced with methods requiring extra saccades that increased the KSPC rate (as in gaze-gesture-based systems), only a minimal speed gain was obtained, if any.

Language models and character and word prediction can provide methods for more efficient text entry. Salvucci (1999) developed advanced methods wherein the user can look around the virtual keyboard and select with direct pointing without a dwell time delay. The system analyzes the gaze path and tries to map the fixations to letters. It uses a dictionary and predefined grammar of how the letters follow each other when deciding whether the fixations belong to the word or not. In the latter case, the user is probably just glancing around in search of the correct letter. The problem with Salvucci's method is that it takes time to deduce whether the fixations belong to a word or not, which is why offline data analysis was performed. Moreover, the accuracy of the system decreases as the number of words in the dictionary increases.

In an experiment with seven novice participants, Salvucci (1999) found that participants “typed” with an average speed of 822 ms per character, which equals about 15 wpm. The typing rates spanned an average of about 28 wpm (430 ms per character) for the fastest participant to about 9 wpm (1272 ms per character) with the slowest participant. The fairly long times needed per character are most probably caused by the long search time needed by a novice who has no previous experience of on-screen keyboards. It would be interesting to repeat Salvucci’s experiment to find out how rapidly true experts can glance from character to character and what their pointing accuracy is. This might give an indication of the fastest text entry speed obtainable by direct pointing.

*ShapeWriter*¹⁶ (previously known as SHARK, for “shorthand aided rapid keyboarding”), developed by Zhai and Kristensson (2003), uses a method similar to Salvucci’s approach with pen (stylus) pointing: the user writes by moving the pointer from one character to another without lifting it. This is comparable to dwell-free gaze pointing where the eye is always on and continuously pointing; it may thus yield an approximation to the potential top speed. In an informal study (Kristensson & Zhai, 2004), two expert users (the authors) were able to write with a record speed of about 50–70 wpm. Evidently, *ShapeWriter* has not yet been tested with gaze pointing. Doing so would be most interesting, since one can assume that gaze pointing would be as fast as, or faster than, hand pointing (provided that the targets are big enough and the eye tracker is accurate enough).

For gaze, the highest entry rates, over 20 wpm, were achieved with Dasher (Ward & MacKay, 2002), which has character prediction built into it. Obviously, if one can write several characters or a word instead of a single character with one stroke, the keystrokes-per-character rate will be lower, which in turn, can speed up writing. In the following section, we briefly introduce common ways to implement word and character prediction in gaze-based text entry systems and related research. We will not go into details of the underlying algorithms and language models; therefore, we will conclude the section by giving some pointers for further reading.

6.2 PREDICTED WORD LISTS

A common way to implement word prediction is to present a list of predicted words for the user. The words are based on the letters the user

¹⁶ For more information on *ShapeWriter*, related publications, and free downloads, see http://www.almaden.ibm.com/u/zhai/shapewriter_research.htm (accessed 1 March 2009).

has written so far. The list is dynamically adjusted as more letters are written and the number of possible continuations of the word decreases. GazeTalk (illustrated in Figure 6.1) provides both letter and word prediction (Hansen et al., 2003b). The six cells toward the bottom right contain the six most likely letters to continue the word that is being entered (shown in the two top left cells). The leftmost cell in the middle row provides shorthand access to the eight most likely completions: with activation of that button, the screen changes into one where those words populate the cells on the two bottom rows. If none of the suggested continuations (words or letters) is correct, the user has access to the button labeled “ABCD...” for populating the bottom-row cells with the next options in the hierarchy.

In Figure 6.1, the user has entered “Gr”; on the basis of those two letters, GazeTalk has predicted a list of potential words: Great, Granted, Greg, etc. Many systems support adaptive learning, meaning that new words written by the user are automatically inserted into the vocabulary and the probabilities for existing words are adjusted on the basis of their usage statistics.

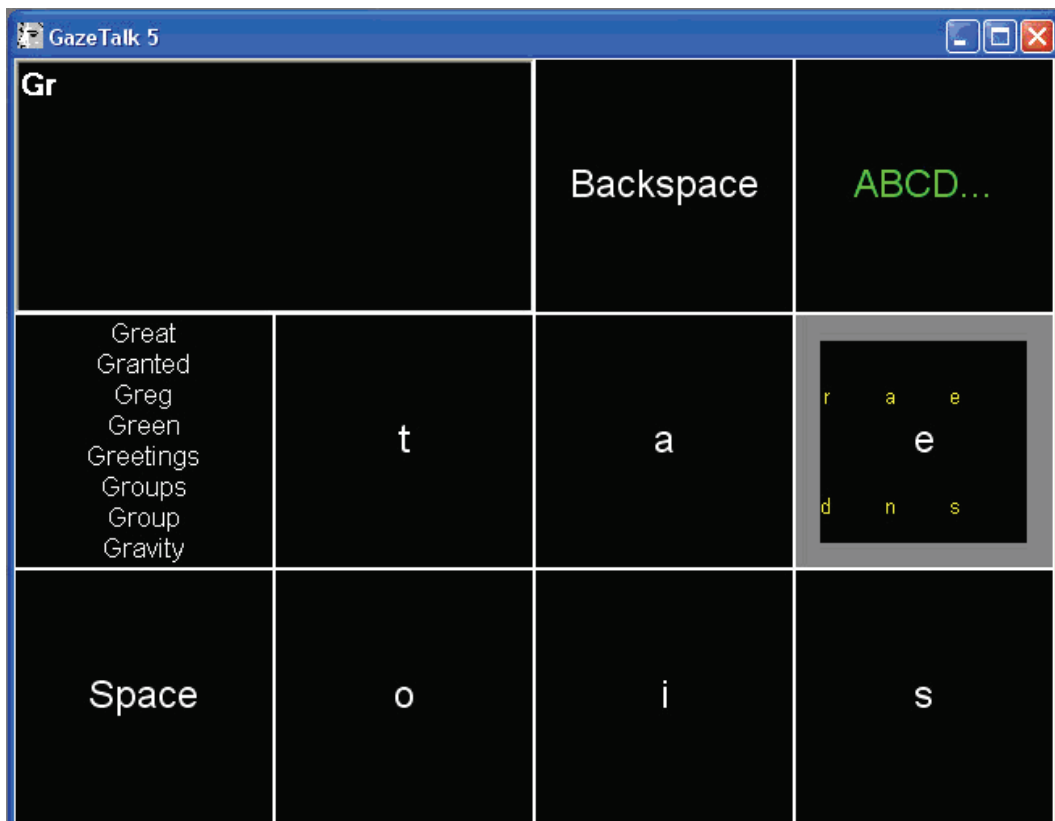


Figure 6.1. GazeTalk provides both word and letter prediction features. GazeTalk’s new interface shows a preview of the next character layout within the cell that is currently being selected. Thus, the user can proceed directly to the correct cell after the current letter has been selected and hence save the search time needed for locating the cell for the next letter.

Word continuations are predicted on the basis of the letters typed so far. After a space, the situation is trickier. A simple way to implement the prediction after a space is to show the next probable word candidates from the most frequent general words such as “the,” “of,” and “an” (MacKenzie & Zhang, 2008). Another solution is to use semantic information from the previous text; for example, “my name” continues with “is” typically.

6.3 CHARACTER PREDICTION

In addition to providing the list of predicted words, GazeTalk uses character prediction to dynamically change the characters shown in the cells that contain the most probable next letters. In Figure 6.1, one of the most likely next letters after “Gr” is “e,” which is being selected in Figure 6.1 (hence the highlighting in its background).

In an experiment by Hansen et al. (2003b), novice participants found the dynamic, predictive layout confusing. Since the order of the letters was constantly changing, they had to search for the desired letter every time the display changed: they found it confusing that the same letter was not always shown in the same location (the same cell). Therefore, the developers decided to set “home positions” for all letters so that they could be found in the same cell unless a letter with higher probability shares the same home.

Recently, the developers of GazeTalk further improved the dynamic layout. They implemented a new interface that shows a preview of the layout within the key that is selected, so that the user can proceed directly to the correct cell when the cells are reoccupied with the letters after the current selection. Note the small yellow letters around the “e” in the selected cell in Figure 6.1 (and compare this to the previous image of GazeTalk, in which no such character preview was available, shown in Figure 5.2). To my knowledge, at the time of writing, this new layout with the character preview has not yet been tested with users, even though it has been publicly available for a while (with free download via the Web).

GazeTalk fills the cells with new (most probable next) letters every time the user types a letter. Another approach is to show all letters at once, by placing several of them in each cell. The user can then type by simply selecting the cell containing the desired letter, and, on the basis of the underlying language model, the program decides which of the possible letter combinations is most likely the word the user wants to write. This approach is similar to the predictive text entry used in cell phones (such as T9). This way, the number of keystrokes required per character can be reduced significantly.

Dasher takes the character prediction feature a step further. Dasher's prediction includes all characters, not only letters (Ward et al., 2000). Thus, common punctuation marks are given more space than rarely used punctuation. With Dasher, the user can select several characters, whole words, or sometimes small phrases all at once. They are also all shown near the current gaze point, so that the user does not need to glance through a separate list.

6.4 THE COST OF THE ADDITIONAL COGNITIVE AND PERCEPTIONAL LOAD

Word prediction, or word completion, is especially useful with highly ambiguous keyboards that have only a few buttons. Such keyboards can provide efficient text entry with low motor or accuracy demands (Harbusch & Kühn, 2003), because, using the word prediction feature, one can reduce the number of keystrokes required to write the word. One should, however, keep in mind the additional cost of perceptual and cognitive load, caused by shifting the focus from the keyboard to the word list and the repeated scanning of the list. Because of the added cost of scanning the list, the actual benefit may be smaller than expected from simply calculating the potential keystroke savings. In some cases, the use of word prediction may even decrease the text entry rate. Koester and Levine (1994b) had both able-bodied participants and disabled ones (suffering from spinal cord injuries) transcribe text with and without the word prediction feature. They found modest enhancements for the able-bodied participants, but the cognitive cost of much slower list search times for the injured participants was so high that in their case the word prediction feature had a negative effect on performance. Therefore, one should carefully consider the implementation and layout. For example, one may want to optimize the list of predicted words to better match the current context or vocabulary of the user, and let the user adjust the number of items shown in the list to match his or her perceptual capabilities and preferences. Trnka et al. (2008) compared two different word prediction methods with a letter-by-letter text entry system. They found that word prediction can improve text entry rates, and that a more accurate prediction system gives better results. This is partly an effect of greater utilization of the prediction feature: if the prediction is accurate, people trust it and use it more. In gaze-based interfaces, the space taken by the word lists is also an issue worth considering; if the words are located within separate buttons, those buttons reserve precious screen space.

MacKenzie and Zhang (2008) compared word and letter prediction in a gaze typing system. Their system (illustrated in Figure 6.2) predicted the next probable words and showed them on buttons located below the text input field. In addition to word prediction, MacKenzie and Zhang experimented with character prediction in an on-screen keyboard. When

the user typed a character, the system highlighted the three most probable next letters on the keyboard (“e,” “i,” and “a” in Figure 6.2). They expected the highlighting to speed up letter selection if the desired letter is one of the highlighted letters, since the search task is reduced from 26 to three characters. Naturally, if the desired letter is not among the highlighted letters, the effect may be negative.

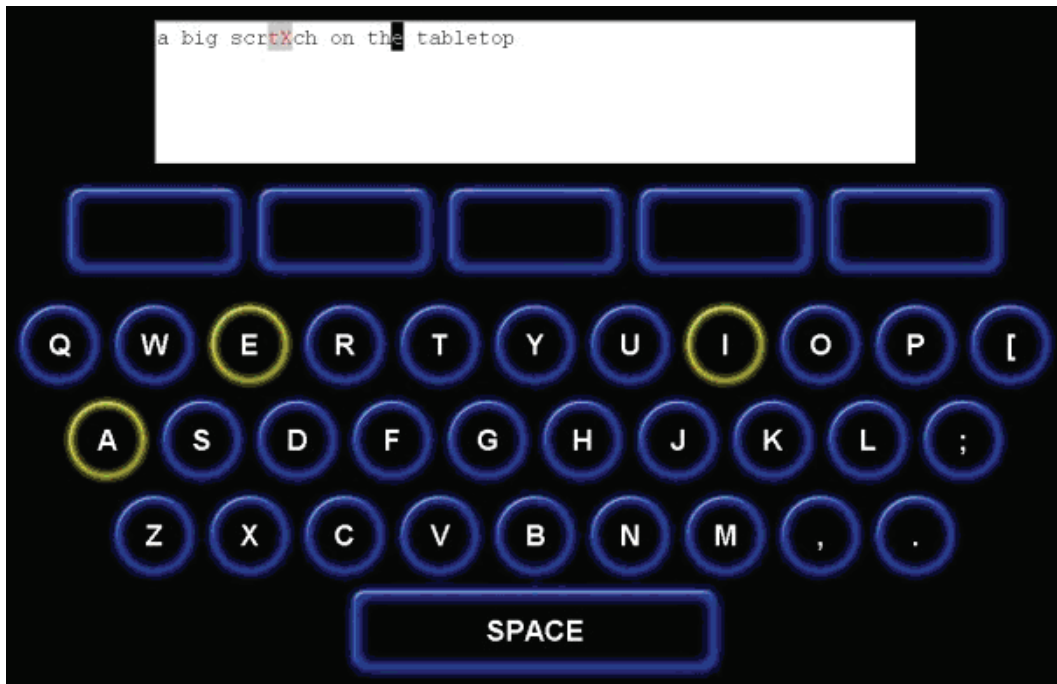


Figure 6.2. Letter prediction is used to highlight the next probable letters on the keyboard. This may help a novice user to find them and thus speed up text entry. The empty boxes between the text input field and letter keys are filled with predicted words if word prediction is enabled (MacKenzie & Zhang, 2008; © ACM, Inc. - reprinted with permission).

In addition to enhancing the GUI via highlighting, MacKenzie and Zhang used the predicted letters for improving the accuracy of their fixation algorithm: they adjusted the measured point of gaze by correcting drift according to the probabilities of letters (keys) near the measured gaze point. For example, if the measured gaze point was located on the border of the key with “d,” the algorithm would still select “e” because of its higher probability.

MacKenzie and Zhang (2008) conducted an experiment with 10 participants to compare the letter and word prediction with two button sizes (small and large). Entry speed ranged from 10.8 wpm to 12.3 wpm. The results show that letter prediction was about 10% faster than word prediction when small buttons were used. With large buttons, word prediction was about 10% faster than with small buttons, probably because the larger size made it easier to recognize the predicted words. With large buttons, there was little or no improvement in entry speed. MacKenzie and Zhang concluded that letter prediction was as good as word prediction, or even better in some cases. Hence, there is potential in

such letter prediction, especially with an unfamiliar layout (the experiment's participants were familiar with QWERTY).

6.5 FURTHER READING

The brief introduction above provided only a general overview of the word and character prediction methods commonly used in gaze-based text entry systems. Below we offer a few starting pointers to related research that may be useful for a reader who wishes to know more about the underlying algorithms, language models, and related issues.

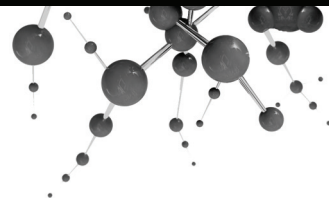
MacKenzie and Tanaka-Ishii's 2007 book *Text Entry Systems: Mobility, Accessibility, Universality* provides a good overview of text entry in general, including chapters specifically on language models and text entry using a small number of buttons, as well as text entry by gaze and text entry by people with disabilities.

The KSPC metric can be used to calculate potential benefits and savings of strokes for word lists. For example, for the list of predicted words, there may be an optimal number of items for the list (e.g., five) after which there is little improvement in KSPC for larger lists. For more information, see MacKenzie (2002) and MacKenzie et al. (2006).¹⁷

One of the first papers to discuss character prediction in the context of gaze-based text entry is by Frey et al. (1990). They calculated the most probable letter pairs in order to better organize the hierarchical menu of a gaze-controlled text entry interface where eye tracking accuracy limitations required that only a few buttons be visible at a time.

Hansen et al. (2003b) discuss natural language processing algorithms, word- and letter-level language models, corpus collection, and adaptive vocabulary as used in GazeTalk. More information about the language models and algorithms used in Dasher can be found in Ward et al. (2000).

¹⁷ I. S. MacKenzie's home page is a good resource for research papers and tools related to methods and measures for analysis of text entry; see <http://www.yorku.ca/mack/> (accessed 1 March 2009).



7 Layout

7.1 COPING WITH INACCURATE TRACKING

The most common method of gaze typing consists of selection of keys from an on-screen virtual keyboard. Typically, only one keystroke per character is needed, since most letters can be directly pointed at and selected. Having all characters visible at the same time requires space. The keys on the virtual keyboard must be big enough to accommodate the accuracy limitations of eye tracking devices.

The inaccuracy of the measured point of gaze was a particularly significant problem in the early days of eye tracking. Therefore, the keys on the screen had to be quite large. For example, the first version of the ERICA system (Hutchinson et al., 1989) had only six selectable items available on the screen at a time. The letters were organized in a tree-structured menu hierarchy. The user selected first a group of letters, then either another group of letters or the single target letter. Typing was slow – it took from two to four menu selections to select a single letter, meaning that several keystrokes were needed for entry of one character. Letters were arranged in the hierarchy on the basis of word frequencies, so that the expected number of steps for text entry was minimized (Frey et al., 1990).

A similar hierarchical method is used by the EagleEyes system (Gips & Olivieri, 1996). It has only two levels (see Figure 7.1) and few special keys. The upper row consists of groups of letters. The letters of the selected group appear in the boxes below the text field. The bottom row of boxes

includes a space key, a key for speaking the text written, and a delete or return key – depending on the state of the program.

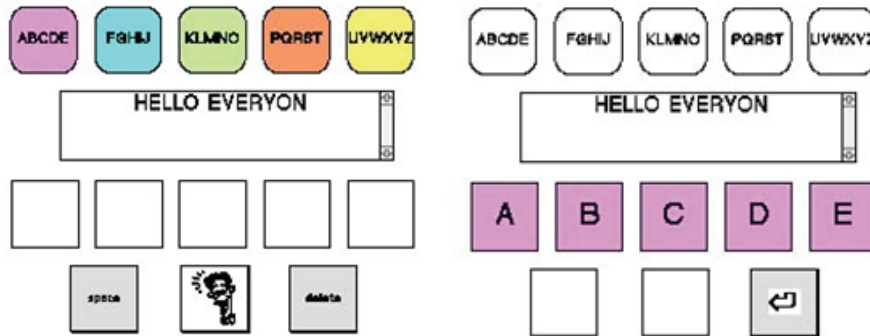


Figure 7.1. Two screens from the EagleEyes two-level speller (Gips & Olivieri, 1996) (reprinted with kind permission from Dr. James Gips).
A new version of the program is available online at <http://www.staggeredspeech.org/>.

Layouts with large keys are still needed and used in today's systems. Some medical conditions cause involuntary head movements or eye tremor, preventing a good calibration (Donegan et al., 2005) or may even restrict eye movements to one direction only. For example, Figure 7.2 illustrates a grid constructed for a person who is able to move his eyes only vertically.

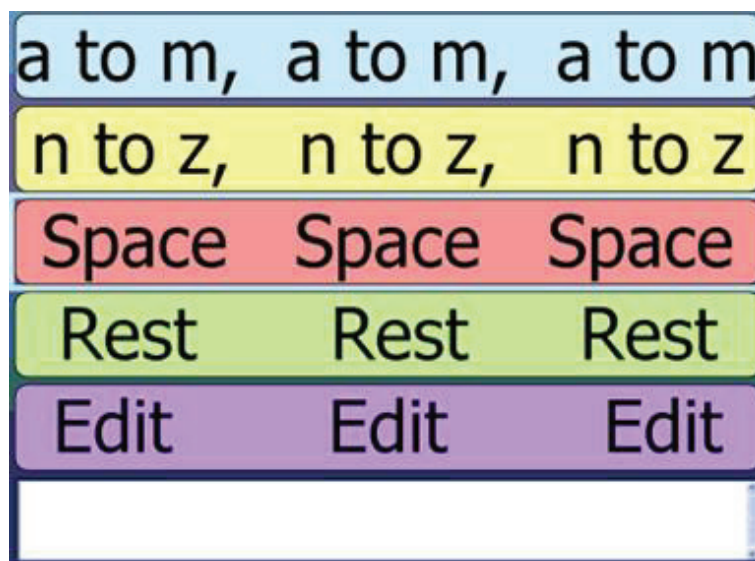


Figure 7.2. Text entry interface for a person who is only able to move the eyes vertically.¹⁸

¹⁸ This grid and a video demonstration of its use are freely available on the COGAIN Web site, at http://www.cogain.org/user_involvement/exemplars/writing-and-computer-control (accessed 1 March 2009).

Furthermore, even though the state-of-the-art eye trackers are fairly accurate (to 0.5–1 degrees), the so-called low-cost systems still do not reach the accuracy levels needed for a QWERTY keyboard. For example, the GazeTalk system (Hansen et al., 2001) was developed with a standard webcam in mind (Hansen et al., 2002). It divides the screen into a 3 x 4 grid (see Figure 6.1). Such big buttons enable easy selection even with inaccurate pointing devices.

7.2 SAVING SCREEN SPACE WITH COMPACT KEYBOARD LAYOUTS

Obviously, if the keyboard occupies most or all of the screen real estate, it significantly limits the space available for other applications. Several attempts have been made to solve the problem of coping with the inaccuracy of the measured point of gaze while still preserving maximum screen space.

Decreasing the number of keys can save screen space. Isokoski (2000) used off-screen targets in order to preserve maximum screen space. Some recent gaze gesture systems use parts of the screen itself as active areas for gesture recognition (Drewes & Schmidt, 2007; Porta & Turina, 2008) or show a small special area where the entry of the gaze gestures occurs (Wobbrock et al., 2008). All these systems save screen space, but learning the gesture-based alphabet takes time. They also require several (typically 2–4) strokes per character. In experiments, users have achieved an average speed of 5–8 wpm (Porta & Turina, 2008; Wobbrock et al., 2008).

Miniotas et al. (2003) developed Symbol Creator, in which a character is created by combining two (or more) symbols (see Figure 7.3). Hence, two keystrokes produce one character (with a few exceptions). The symbol parts and their combinations resemble handwritten characters or portions thereof (for instance, as “o” and “l” put together form “d”), which aids in learning the symbols. Symbol Creator has eight keys in a one-row virtual keyboard. Showing only one row of keys leaves most of the screen free for other purposes. The authors of the study reported an average typing speed of 8.5 wpm in the experiment’s last session.

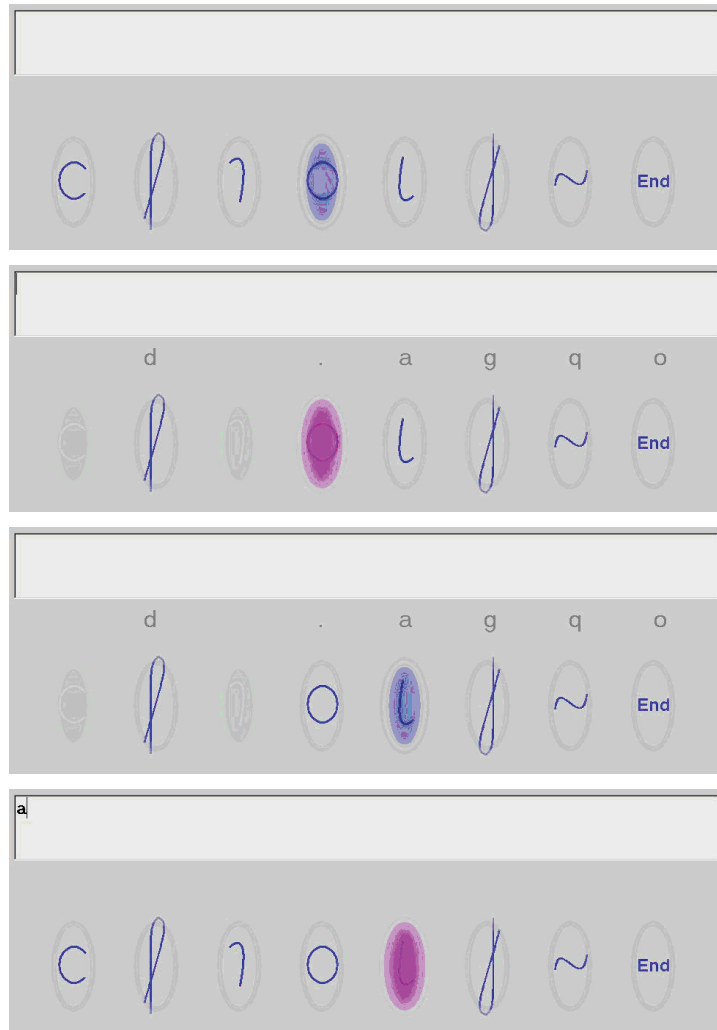


Figure 7.3. The letter “a” is written with Symbol Creator (Miniotas et al., 2003) by combining two symbols. First the user focuses on “o” - which is indicated with the blue highlighting (topmost image). Selection of the first part (“o”) is confirmed with the red highlighting and the system gives hints for the next part (second image from top). The second part of the symbol is then focused on (third image), highlighted, and finally typed into the text input field (last image).

7.3 SCROLLABLE KEYBOARDS

Our (Špakov & Majaranta, 2008) goal was to develop a keyboard that saves screen space but is still immediately usable and does not require any special learning. Our idea is to use a keyboard layout that is already familiar to the user (such as QWERTY) and to save screen space by showing only part of the keyboard. The familiarity of the keyboard layout significantly affects learning time when a new input method is used, because of skill transfer (MacKenzie et al., 1999). It should be noted that QWERTY may not be the best choice for people with disabilities who have no previous experience with the QWERTY layout and might thus find another kind of layout (for example, an alphabetically ordered layout) more familiar.

For the “full” keyboard, we used the QWERTY layout, a common keyboard layout, shown in Figure 7.4 on top. For the experiment, we decided to omit special characters and punctuation (other than the comma and period keys). Two space keys were used, at the end of the second and the third row.

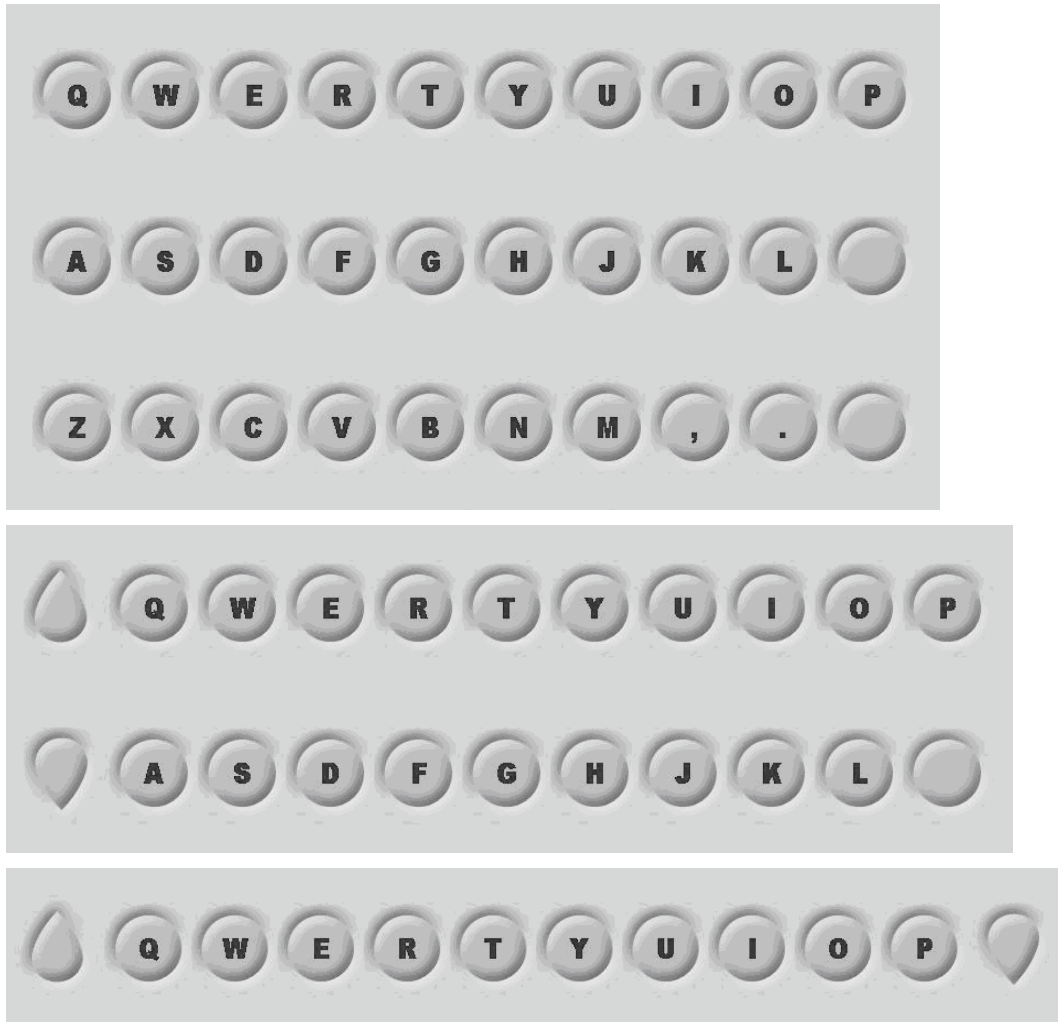


Figure 7.4. Full (three-row) keyboard, two-row keyboard, and one-row scrollable keyboard.

The two-row keyboard (Figure 7.4, in the middle) has only two rows of keys visible at any given time. To reach the third row, the user needs to select one of the special scroll keys on the left. The one-row keyboard (Figure 7.4, on the bottom) shows only one row. The scroll keys – for up and down – are located on the sides of the keyboard. In both, the scrolling is cyclic; an invisible row can be reached by using either of the scroll buttons. The scrolling produces animated feedback, which takes 150 ms. Obviously, the KSPC figure is more than 1 for the scrollable keyboard, since at least one extra keystroke (use of the scroll key) is required to reach a hidden row.

The visible distance between rows was extended because the drifting of the measured gaze position is higher vertically than in the horizontal

direction with the tracker we used (see the section on methods, below). Even though the visible buttons are circles, the gaze-reactive area for each button is a rectangle (approximately 1.5 by 3.0 degrees if the distance between the user and the monitor is 45 cm) that covers the whole area between the visible buttons. The buttons were selected with a dwell time of 500 ms, which remained constant throughout the experiment. Animated feedback indicated the progression of the dwell time, and the key became “pressed” (shown as pressed “down” for 150 ms) when selected.

Method

We conducted an experiment to study the efficiency and usability of the scrollable keyboard. In the experiment, we compared the full three-row keyboard to the two layouts of the scrollable keyboards (two- and one-row) illustrated in Figure 7.4. Eight volunteers (aged 23–47 years, five male and three female) took part in the test. They were students or staff at the University of Tampere, and all had participated in other eye typing experiments. Experienced participants were used in order to minimize the learning period. All were fluent in English and familiar with the QWERTY layout.

The experiment was conducted in the usability laboratory of the University of Tampere. A head-mounted EyeLink eye tracking system was used to measure participants’ eye movements. The iComponent software, which has a plug-in for EyeLink, was used to implement the experimental keyboard and to save data. The setup consisted of operator and subject monitors, adjustable chairs, and tables. The chair was set such that the participant’s eyes were approximately 45 cm from the 17-inch monitor.

For the experiment, 30 easy-to-memorize phrases were chosen from a set of 500 phrases proposed by MacKenzie and Soukoreff (2003). Punctuation was removed, and the phrases were case-insensitive. Participants were instructed to eye type the phrases as rapidly and accurately as possible. They were instructed to ignore mistakes and to carry on with a phrase when a mistake was made (our keyboards did not have a backspace key).

Each session started with a short training period on the two-row keyboard. To provide a basic level of familiarity with the experimental software, participants were given one practice phrase (about 25 characters) prior to data collection.

The experiment used a (one-way) repeated measures design with three conditions: a three-row (full), two-row, and one-row keyboard. There were eight sessions, each including all three testing conditions (one session per day). The order of conditions within the same session was counterbalanced between participants. Each session included six phrases (average length: 26.3 characters) for each condition, shown one at a time.

Thus, the number of characters entered was approximately $8 \times 8 \times 3 \times 6 \times 26.3 \approx 30,300$ (1,152 phrases). A session lasted approximately 10–15 minutes.

Results

The results for the last session show an average typing speed of 15.06 wpm for the full keyboard, 11.12 for the two-row keyboard, and 7.29 wpm for the one-row keyboard (see Figure 7.5).

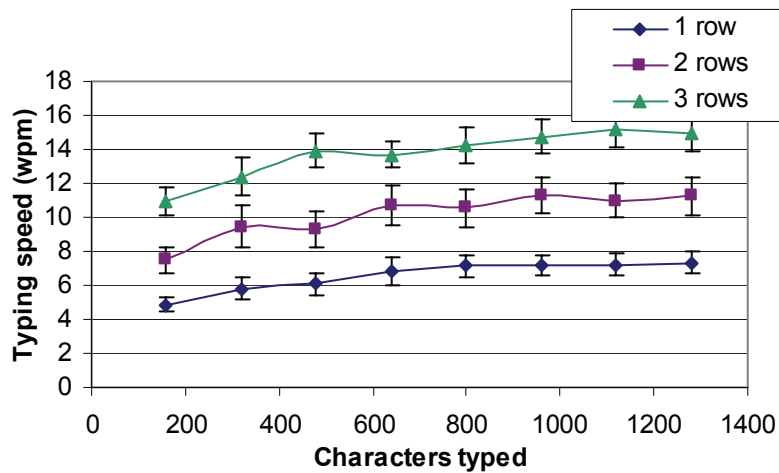


Figure 7.5. Average typing speed in words per minute and error bars for the eight sessions.

The average error rates varied by 1–5%, with large variance between participants over the whole experiment. In the last session, the average error rates were below 2% for all conditions (see Figure 7.6).

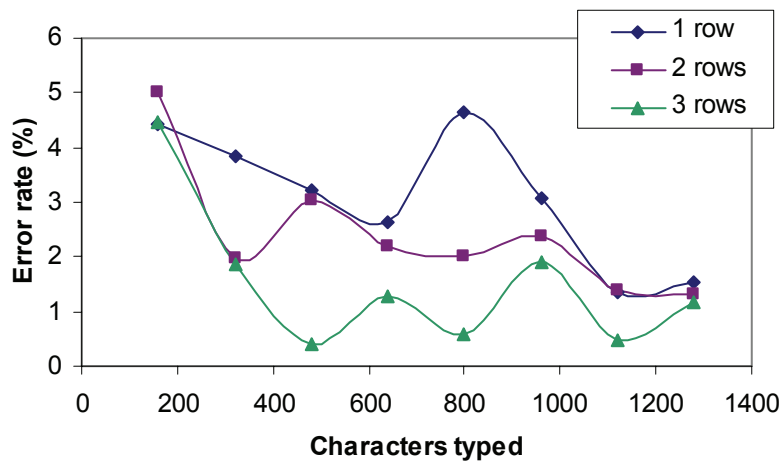


Figure 7.6. Error rate (%).

The selection time for the scroll buttons, letter keys, and space character was measured. Monitoring the usage of the scroll button proved especially interesting, because it shows how the participants learned to use the scrollable keyboards with only partially visible layout. Figure 7.7 shows

the selection times for the one-row (top) and two-row (bottom) keyboard. The decreasing values for the scroll buttons' selection time in both graphs during the first five sessions show the approximate amount of text one was required to type (~1,000 characters) in order to learn this input technique. The average selection times of the scroll buttons in the last (eighth) session were 1107 and 1268 ms for the one-row and two-row keyboard, respectively. These values are still higher than the letter buttons' selection times (1016 and 961 ms), especially in the case of the two-row keyboard.

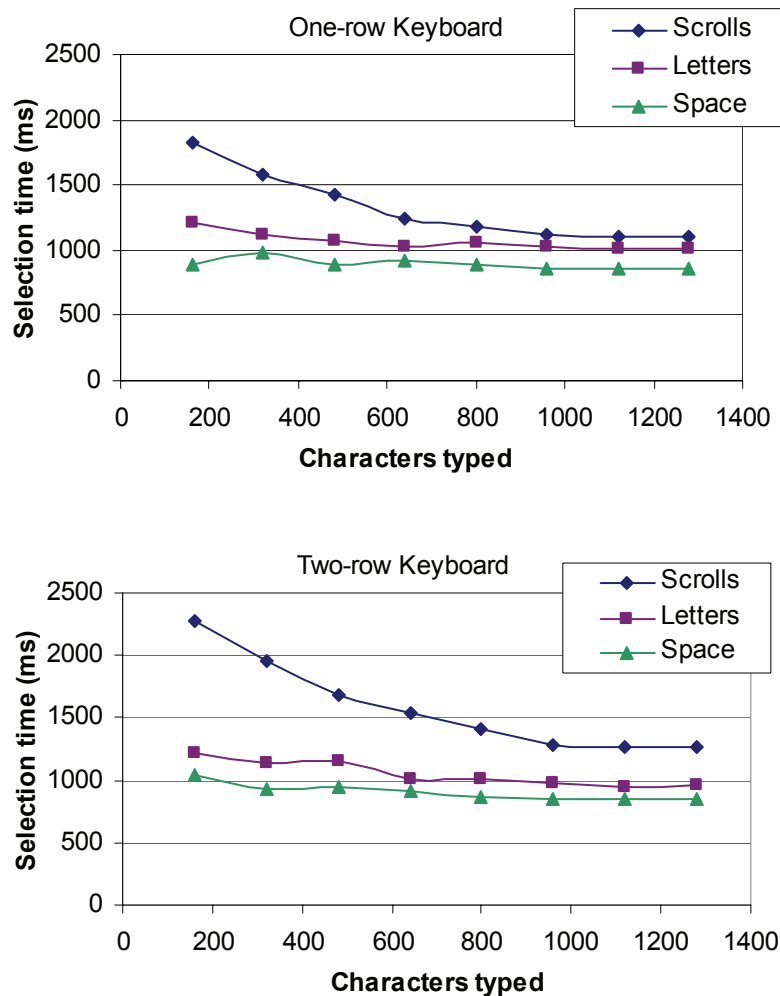


Figure 7.7. Selection time for the one-row (above) and two-row (below) scrollable keyboard.

Analysis of the scroll button usage shows that it decreased slightly with time and the average percentage of scroll button clicks among all clicks was 39% (1.64 KSPC) and 16.5% (1.2 KSPC) for the one-row and two-row keyboard, respectively.

Participants used different strategies with the scrolling keyboards. Half of them memorized the location of letters and rows so that they could choose the shortest route to the invisible row and thus minimize scroll button usage. For example, after "e" (located on the top row), the user can reach

“n” (on the bottom row) with one scroll up instead of two scrolls down in the one-row keyboard. Some participants never scrolled the layout from top line up (to the bottom) or vice versa, because they did not want to lose orientation in scrolling. In this case, more scrolling was required but the participants did not spend time searching for the target letter. Finally, one participant did not memorize the distribution of letters across rows, always visually scanned the rows to find the desired letter, and used only one direction of scrolling (up). This strategy resulted in the slowest typing speed. The difference between the fastest and slowest participant was approximately 3 wpm within each condition.

Redesign: Layout Optimization of the Scrollable Keyboard

Analysis of the usage of the scrolling buttons revealed that the keyboard could benefit from optimization of the layout (for details, see Špakov & Majaranta, 2009). Our optimized layout was created on the basis of the assumption that usage of the scroll buttons would be reduced by grouping the most frequent letters on the same row. The most frequent letters were placed in the first row, the least frequent letters in the last row, and the (most frequently used) space button in each row (we removed the comma button). The spatial distribution of the letters in the same row is based on the digram analysis: it is optimized such that the length of the gaze path is minimized within the row (keeping the position of the space button fixed – it is always the rightmost key).

We tested the optimized layout in an experiment that followed the method and procedure of the first experiment. The only difference was that the condition with a full-sized keyboard was omitted since we assumed that the typing speed would be the same after participants learn the layout. The results show an average typing speed of 8.86 wpm for the one-row keyboard and 12.18 wpm for the two-row keyboard. Error rates remained at a level of approximately 2%, independently from the keyboard.

Typing with the optimized layout required less scroll button usage than the reduced QWERTY layout did. The scroll button selections produced 33% (1.49 KSPC) of all clicks with the one-row keyboard and 10% (1.11 KSPC) with the two-row keyboard. The usage of the scroll buttons remained at roughly the same level across all sessions.

Discussion

As expected, the optimized layout was initially harder to use, because of the unfamiliar distributions of letters. However, the results show that the optimized layout did indeed improve typing efficiency by decreasing the usage of the scroll buttons: 33% versus 39% with the one-row keyboard (an 18% reduction) and 10% versus 16.5% for the two-row keyboard (a

40% reduction). The reduction in the frequency of scroll button usage helped to increase the typing speed from 7.26 to 8.86 wpm (increased by 22%) for the one-row keyboard and from 11.17 to 12.18 wpm (increased by 9%) with the two-row keyboard.

Since every third click is produced by the selection of a scroll button in the optimized one-row condition, the over-production rate caused by the scrolling is 1.49 KSPC. In typing with the optimized two-row keyboard, every tenth click is produced over a scroll button, with a rate of 1.11 KSPC. These keystroke rates are quite reasonable when compared to the figure for direct pointing with a fully visible keyboard: an optimum of 1 KSPC.

With both keyboards, the scrolling was cyclic, so that the users could scroll the keyboard around both ways. Even though this is considered efficient, especially for the one-row keyboard, since the user can always select the shortest route (one scrolling action) to the desired key, it may be confusing for some users who want to maintain the orientation of the layout. Thus, for some users it might be useful to provide an option to prevent scrolling from the first (topmost) row to the third (bottommost) row. Furthermore, if the feedback on the scroll button were to reflect this constraint (e.g., by indicating a “disabled” mode), it might help the user to maintain the orientation within the partly shown (partly hidden) keyboard. The scroll keys might also be easier to hit with a single saccade if they were more peripherally salient – this could be achieved by adding arrow icons on the keys (which might also show the direction of scrolling more clearly than the shaped buttons alone do).

Further improvements might be possible with the introduction of character or word prediction. If the visible row were dynamically constructed on the basis of the text written so far such that it always showed the most probable next letters, the typing might speed up, in theory. However, as discussed in Chapter 6, users may find it confusing if the layout changes dynamically and the added perceptual load and cognitive cost may counteract the benefit.

Conclusion

We have shown that scrollable keyboards, which reduce the space taken by the full (three-row) keyboard by 1/3 or 2/3, can be efficiently used to enter text by gaze. Typing speed fell by only 51.4% for the one-row and 25.3% for the two-row keyboard from the speeds seen with the conventional QWERTY layout. Furthermore, the increase in the rate of keystrokes was quite reasonable, from 1 KSPC to 1.64 and 1.2 KSPC with the one-row and two-row keyboard, respectively.

By optimizing the keyboard layout according to the letter-to-letter probabilities, we were able to reduce the frequency of scroll button usage,

which enabled a further increase in the typing speed, from 7.26 wpm (with the QWERTY layout) to 8.85 wpm (with the optimized layout) for the one-row keyboard and from 11.17 (QWERTY) to 12.18 wpm (optimized) for the two-row keyboard. The results are encouraging in comparison to, for example, gesture-based interfaces that require several strokes per character (although the saccades needed to perform such eye strokes can be very fast).

Scrolling keyboards may be especially useful in casual typing situations, such as filling in Web forms where the overview of the full Web page is important. Scrolling could also be useful in accessing the key rows that are not needed as often as letters, such as number, punctuation, and function keys. Finally, the user should be able to adjust the number of visible rows easily to support the optimal layout in each situation.



8 Feedback

Appropriate feedback is especially important when the same modality is used for both control and perception. When gaze is used to control an application and select objects on the screen, gaze is engaged in the input process: the user needs to look at an object to select it. This means that the user cannot simultaneously control an object and view the effects of the action, unless the effect appears on the object itself. For example, if the user is entering text by gaze, he or she cannot see the text appear in the text input field while simultaneously selecting a letter by “eye pressing” a key on an on-screen keyboard. To review the text written so far, the user needs to move the gaze from the on-screen keyboard to the typed text field. This looking back and forth can become excessive, especially as novices in particular often shift their gaze between the keyboard and the text input field to review the text written so far (Bates, 2002). This shifting can be reduced by adding auditory feedback, such as an audible “click” or pronunciation of each letter as it is written. Experienced users learn to cope with having to use the same modality for input (control) and output (feedback); they complete the task (e.g., scrolling) before gazing at the results (Bates, 2002).

There is a fundamental difference in using dwell time as an activation command when compared to, for example, a button click. When manually clicking a button, the user makes the selection and defines the exact moment when the selection is made. Using dwell time, the user only initiates the action; the system makes the actual selection after a predefined interval. When physically clicking a button, the user also feels and hears the button “click.” Such extra confirming (auditory or tactile) feedback is missing when an “eye press” is used to click, so it must be provided by the application.

This chapter summarizes the results of three experiments studying various aspects of feedback during eye typing. First, examples of the relevant research on feedback are reviewed. The methods and results of the experiments are then presented, followed by guidelines gleaned from the results.

8.1 RELATED RESEARCH

It is known that interaction in conventional graphical user interfaces is enhanced by adding sound (Gaver, 1989), an example being the beep used with warning dialogs in Windows. According to Brewster and Crease (1999), the usability of standard graphical menus is improved by adding sound. In particular, combining visual and auditory feedback is claimed to improve performance and reduce subjective workload, as compared to visual feedback alone.

Non-speech sound also supports scanning as an input method. Brewster et al. (1996) showed that auditory feedback supports the scanning rhythm, helping users anticipate the correct time to press a switch for selection.

Added auditory feedback is also useful in gaze-based interfaces. It can confirm successful execution of a command or notify of a change of mode. For example, Hornof and Cavender (2005) provided both visual and auditory feedback in their EyeDraw program to indicate when the mode changed from looking to drawing. The appearance of the cursor changed, and a sound of a different pitch was played for each transition.

Animation is another way to enhance visual feedback, with progress bars as a typical example. Animation can, for example, help to clarify the meaning and purpose of an icon (Baecker et al., 1991). Furthermore, a shift between two conditions is easier to understand if the change is animated. For example, in Cone Trees by Robertson et al. (1991), changes in 3D trees are animated (e.g., with rotation or zooming). Animation allows the perceptual system to track changes in perspective.

Velichkovsky and Hansen (1996) suggest four different generic formats for presenting items that can be selected by gaze but have not yet been selected (thus indicating items that can be interacted with): 1) a traditional on-screen button, 2) a frame or a “halo” around the selectable object, 3) changing of the detail level such that selectable items are shown in more detail and background items blurred, and 4) no visible areas being shown but objects still being able to react to gaze. The first option is obvious for desktop applications where it is important that the user immediately understand which objects can be commanded. The second and the third option, framing an object and blurring its background, are used in, for instance, video games to indicate which items can be interacted with. The

last option, with no explicit feedback, is most suitable for non-command (Nielsen, 1993) or attentive interfaces where the user is not expected to change the gaze behavior to give explicit commands but the information of the user's natural eye movements is used subtly in the background. For example, Vesterby et al. (2005) suggest gaze-guided viewing of interactive movies where the plot of the movie changes according to the viewer's visual interest. If the user is explicitly required to give commands and if gaze-reactive areas are emphasized by explicit feedback, it might disturb the immersion and the viewer might lose track of the story line. The level of feedback required depends on the application and task. Hyrskykari et al. (2003) note that even in attentive applications it is important to provide enough feedback that the user does not lose control and is able to react to potential problems caused by the inaccuracy of gaze. This chapter focuses on only the first option: how to show proper feedback on the on-screen buttons of a virtual keyboard, used to enter text by gaze (using dwell time for the activation command). In particular, the discussion will address the feedback given on *focus* (the focused item is pointed at by gaze) and *selection* (the item is selected by gaze).

Well-known guidelines (e.g., Microsoft Windows User Experience, 2002) suggest that continuous feedback should be used for continuous input (e.g., moving a cursor, dragging an object), and discrete feedback for discrete input (e.g., highlighting the selected object). In eye typing, the action is a combination of continuous and discrete input. The user controls a visible or invisible cursor by moving the gaze (continuous input). When dwell time is used as the activation command, the user fixates at the desired target and waits for the action to happen. The typing action itself is a discrete selection task. Since eye typing requires both continuous and discrete actions, choosing the proper feedback is an interesting design issue.

Seifert (2002) studied feedback in gaze interaction by comparing 1) continuous feedback using a gaze cursor, 2) discrete feedback by highlighting the target under focus, and 3) no feedback for the gaze position. Seifert found no differences in performance between the gaze cursor and the highlight conditions. However, the condition with no visible feedback led to significantly shorter reaction times, fewer false alarms, and fewer misses. In Seifert's study, there were only three (large) letters displayed at a time. In eye typing, in use of the QWERTY keyboard layout with considerably smaller on-screen targets, having no feedback is not a realistic option, since it requires a very accurate eye tracker. There are eye typing systems that display only a few large keys and do not require such accuracy (such as GazeTalk, shown in Figure 8.1). They often use intelligent word prediction methods. In this study, the QWERTY keyboard layout and no word prediction have been used in order to keep the experimental setup as simple as possible.

It is surprising that Seifert found no performance differences between the (continuous) cursor and the (discrete) highlight condition, since previously it was assumed that the constant movement of a gaze cursor distracts the user (Jacob, 1995). The distraction is compounded by problems with calibration, which cause the cursor to gradually drift away from the focus of attention. This is especially disturbing in a situation where the user needs to place the cursor in a specific location – for example, to define a starting point for drawing. A solution suggested by Hornof et al. (2004) is to show a grid, which provides visual anchors that the user can fixate on (while ignoring the drifting cursor). To minimize the potentially disturbing effects of showing the gaze point, it was decided not to show the cursor in the current study.

Animation is also exploited in gaze-aware systems. For example, EyeCons (Glenstrup & Engell-Nielsen, 1995) show an animation of a closing eye to indicate dwell time progress. However, EyeCons may be inconvenient in eye typing, if they divert the user’s attention from the target letter.

The ERICA system (Hutchinson et al., 1989) uses animation in eye typing by showing a shrinking rectangle to indicate the progress of selection (Lankford, 2000). First, a key is highlighted by drawing a rectangle around the key. After indicating focus, the rectangle starts to shrink. The key is selected at the end of the shrinking process. A similar approach is used in GazeTalk, illustrated in Figure 8.1. The research presented here includes a modification of this approach. Instead of a rectangle, a shrinking letter is used – the letter on the key. Through shrinking of the symbol itself, the feedback is further simplified. Since motion is an effective pre-attentive feature of vision to guide attention (Hillstrom & Yantis, 1994), it can be hypothesized that a shrinking letter draws the attention toward the center of the key.



Figure 8.1. GazeTalk provides visual feedback on the dwell time progress for the letter “e.”

Showing the feedback on the center of the focused item, rather than the actual (potentially slightly inaccurate) position of the gaze, seems to be especially useful for some users (Donegan et al., 2006b). When the feedback is shown at the center of a gaze-responsive button, the calibration appears perfect to the user, encouraging the user to feel confident when using gaze.

8.2 METHODS AND PROCEDURES

Three experiments were conducted to study the effects of feedback on eye typing speed, accuracy, gaze behavior, and user experience. The feedback was varied in each of the experiments. This section starts with a brief description of the setup and procedure, since they were basically the same for all experiments, followed by an introduction to the metrics used in the experiments. The experiments and the results are then presented in more detail.

Two computers were used along with an iView X RED-III eye tracking device from SensoMotoric Instruments (Berlin, Germany). The eye tracker and the user's screen are illustrated in Figure 8.2. The eye tracker samples at 50 Hz with one-degree gaze position accuracy. The eye tracking device automatically compensates for (slow) head movements. The eye tracker device was placed in front of the corner of the monitor.



Figure 8.2. Experimental setup: on-screen keyboard and eye tracking device.

One of the computers (Subject PC, with 17" flat LCD monitor, 1280 x 1024 resolution) was used to run the experiment and the other (Operator PC) to collect the eye movement data. After real-time transfer of the eye coordinate data from Operator PC to Subject PC, the system saved the data in three separate log files, thus: 1) raw data and 2) fixation data from the eye tracking device, and 3) event data logged by the experimental software. We did not exploit fixations in our software; we calculated the gaze position directly from filtered raw data points. Filtered points were

mapped to screen coordinates, and blinks and erroneous data had been removed automatically by the software, using predefined thresholds.

The experimental software had an on-screen keyboard, a “Ready” key, a “Del” key, and two text fields (one each for the source and typed text – see Figure 8.3). The Finnish speech synthesizer Mikropuhe (v. 4.2), by Timehouse Oy, was used for spoken feedback (with default parameters).

For all experiments, the task was to type short phrases of text. Participants were instructed to first read and memorize the source phrase and then to eye type it as quickly and accurately as possible.

The participant sat in front of the monitor, with a distance of 70–80 cm between eyes and tracker. The participants were instructed to sit still. However, their (head) movements were not restricted in any way. The eye tracker was then calibrated (and, if necessary, re-calibrated) before phrases were shown. Some practice phrases were then entered.

During the experiment, each participant was presented with short, simple phrases of text, one at a time. All phrases were in Finnish, the native language of the participants in all experiments. After typing the given phrase, the participant looked at the Ready key to load the next phrase.

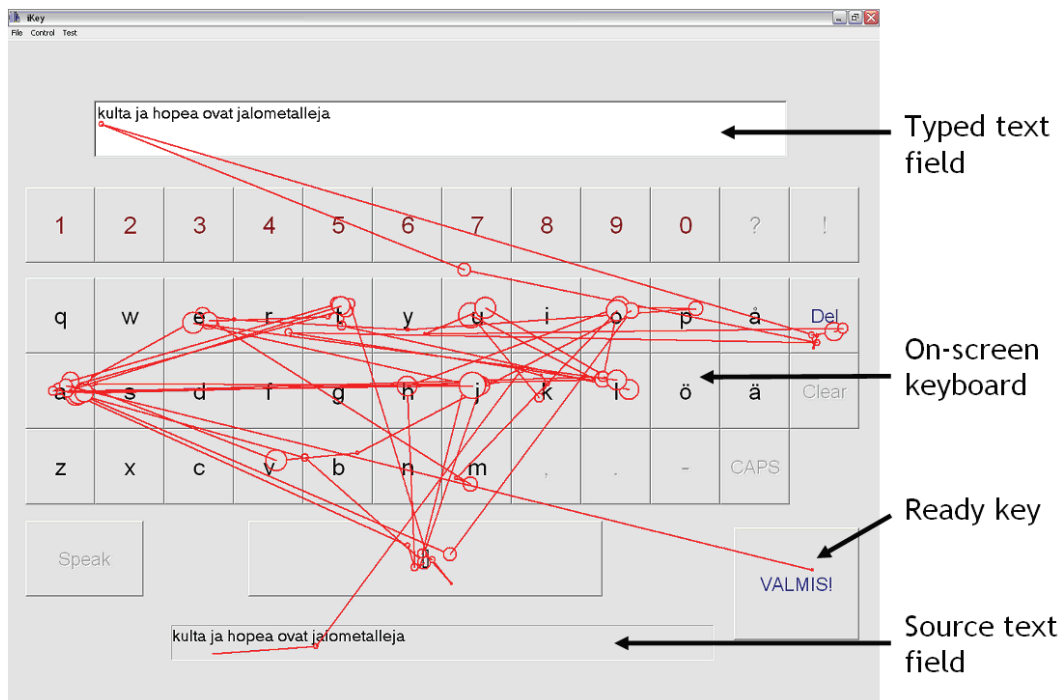


Figure 8.3. An example gaze path for a participant eye typing one phrase.

Participants could correct errors – delete the last letter typed – by looking at the Del key. They were told to correct errors if noticed immediately but not to correct errors in the middle of a phrase if they noticed them after the

entire phrase had been typed. In the analyses, both corrected errors and errors left in the final text are considered.

Measures used in analyzing the results are described in detail in Section 1.2. In summary, the typing speed was measured in wpm. Accuracy was measured by MSD error rate and KSPC. In addition, we used read text events for measuring how often the participant reviewed the text written so far. Participants' subjective impressions were collected with questionnaires and interviews.

The statistical analyses were done using repeated measures ANOVA and Bonferroni-corrected t-tests. Data collection for a phrase started on the press of the first character and ended on the press of the Ready key ("press" in this context refers to successful selection of the key by gaze). Each experiment and the results are reported upon in detail in the following sections of the chapter.

8.3 EFFECTS OF AUDITORY AND VISUAL FEEDBACK

The first experiment (Majaranta et al., 2003a) used a relatively long (900 ms) dwell time in studying the effect of auditory feedback on user performance. Speech and non-speech auditory feedback, as well as no auditory feedback, were tested. The initial hypothesis was that added auditory feedback would improve performance.

Participants and Design

Sixteen participants volunteered for the experiment. Data from three participants were discarded because of technical problems. In the end, there were five females and eight males (mean age: 23 years). All were able-bodied university students with normal or corrected-to-normal vision. None had experience with eye tracking or eye typing, but all were familiar with desktop computers and the QWERTY keyboard layout.

Four feedback modes were tested (see Table 8.1):


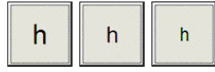

Visual Only. The key is highlighted upon focus and its symbol shrinks as dwell time progresses. On selection, the letter turns red and the key is pressed.

Click+Visual. The Click+Visual method is the same as the Visual Only mode, with the addition of a short audio "click" on selection.

Speech+Visual. The Speech+Visual method is the same as the Visual Only mode except that it also has synthetic speech feedback. The letter on the key is spoken upon selection.

Speech Only. The Speech Only mode does not use visual feedback. The symbol on the key is spoken on selection.

Table 8.1: Feedback modes in the first experiment

Feedback mode	While in focus	When selected
Visual Only 	highlight, shrinking 	red letter, key down 
Click+Visual	highlight, shrinking	red letter, key down, "click"
Speech+Visual	highlight, shrinking	red letter, key down, letter spoken
Speech Only	none	letter spoken

The dwell time for selection was the same, 900 ms, for all modes, including Speech Only. For the "Visual" modes, the 900 ms consisted of a delay before the onset of shrinking (400 ms - indicating focus) and the shrinking itself (500 ms - indicating the progression of dwell time). After the full 900 ms dwell time had elapsed, the selected letter was typed into the input field, "typed text field."

The experiment was a repeated measures design with four feedback modes and four sessions. Participants visited the laboratory four times. Each of the four sessions contained four test blocks, each with a different feedback mode (in randomized order). A block involved the entry of five short phrases of text. There was a pause after each block, and then the participant continued to the next test block with another feedback mode. Thus, each participant typed with all different feedback modes in every session. In the last session, the participants were interviewed and they filled in a questionnaire. The results are based on, in total, 1,040 phrases (13 participants x 4 sessions x 4 feedback modes x 5 phrases).

Results

Typing Speed

The grand mean value for typing speed was 6.97 wpm. This is quite typical for eye typing (Frey et al., 1990; Majaranta & R ih a, 2002) but is too slow for fluent text entry. However, the experiment showed that participants improved significantly with practice over the four sessions ($F_{3,36} = 10.92, p < 0.0001$, see Figure 8.4).

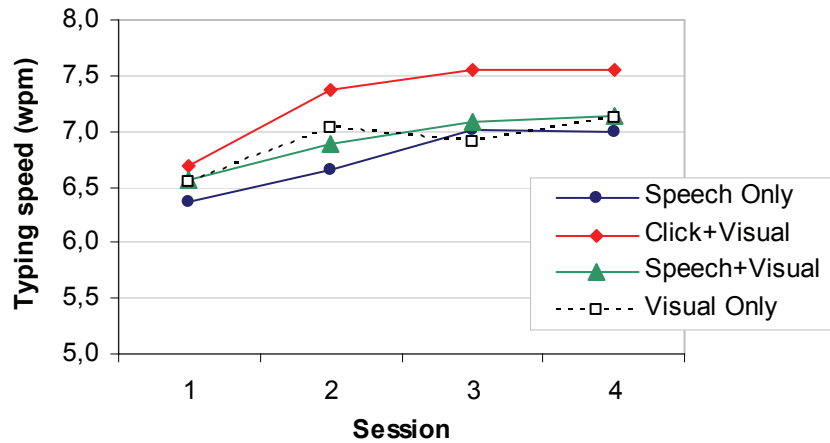


Figure 8.4. Typing speed (wpm) by feedback mode and session.

Feedback mode had a significant effect on text entry speed ($F_{3,36} = 8.77$, $p < .0005$). The combined use of Click+Visual feedback yielded the highest entry rate, with participants achieving a mean of 7.55 wpm in the last (fourth) session. The other fourth-session means were 7.14 wpm (Speech+Visual), 7.12 wpm (Visual Only), and 7.00 wpm (Speech Only).

Accuracy

The mean character level error rate was quite low (0.54%), and the participants' accuracy also improved significantly with practice ($F_{3,36} = .09$, $p = .005$). A significant main effect of feedback mode on error rate was found ($F_{3,36} = 5.01$, $p = .005$). Surprisingly, eye typing with Speech Only feedback was the most accurate technique throughout the experiment, with error rates under 0.8% in all four sessions (see Figure 8.5). Visual Only had the highest mean error rate (0.95%).

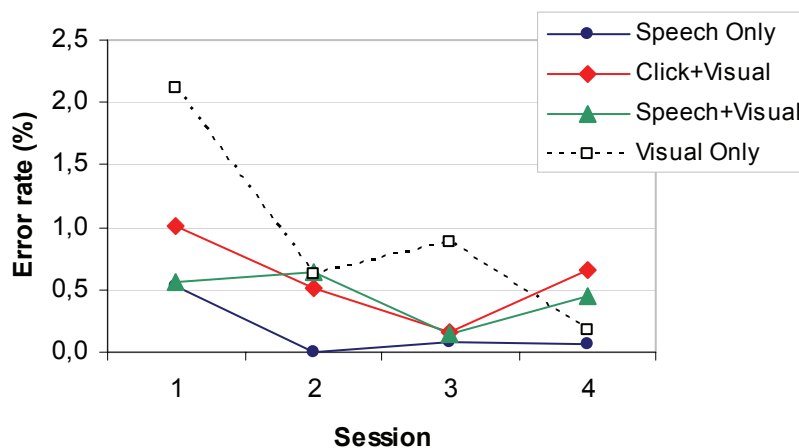


Figure 8.5. MSD error rate (%) by feedback mode and session.

While the very low error rates overall seem encouraging, accuracy is also reflected in KSPC. Before presentation of the results for KSPC, an additional comment on eye typing interaction is warranted. In eye typing, users frequently make errors (especially with short dwell times) and immediately correct them. Thus, measuring accuracy only in terms of errors in the final text is insufficient. On the other hand, a KSPC figure of, for example, 1.12 reflects about a 12% keystroke overhead, due to the errors committed and corrected. Of this, 6% is for the initial error and 6% is for activating the Del key. Thus, KSPC = 1.12 is roughly equivalent to a 6% error rate.

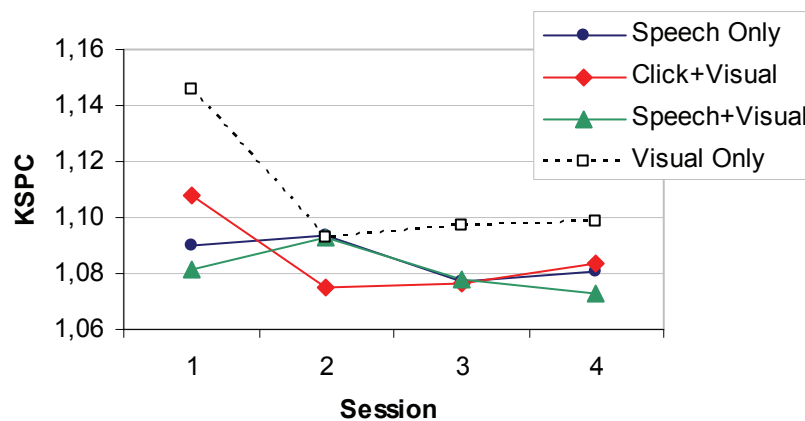


Figure 8.6. KSPC by feedback mode and session.

The grand mean KSPC was 1.09, meaning there was a roughly 9% keystroke overhead in correcting errors (roughly corresponding to a corrected error rate of 4.5%). The KSPC for the Visual Only feedback mode was the highest in all four sessions, ranging from 1.15 for the first session to 1.10 in the fourth session (see Figure 8.6). The effect of feedback mode was significant ($F_{3,36} = 3.60, p < .05$).

Gaze Behavior

No significant differences were found between the feedback modes in the total number of fixation events. However, there were significant differences in the participants' gaze path behavior, measured in the number of times the participant reviewed the text written so far.

In the first experiment, the grand mean was 0.064 read text events per character. By feedback mode, the RTE means were 0.047 (Speech Only), 0.051 (Click+Visual), 0.049 (Speech+Visual), and 0.110 (Visual Only). The differences were statistically significant ($F_{3,36} = 30.06, p < .0001$). In particular, RTE for Visual Only feedback was more than 100% higher than for any other mode (see Figure 8.7). Participants moved their point of gaze to the typed text field approximately once every 10 characters entered for the Visual Only feedback mode but only about once every 20 characters

for the other modes. This may be because auditory feedback (used with all except Visual Only mode) significantly reduces the need to review and verify the typed text and brings a sense of finality that simply does not surface, at least to the same degree, through visual feedback alone.

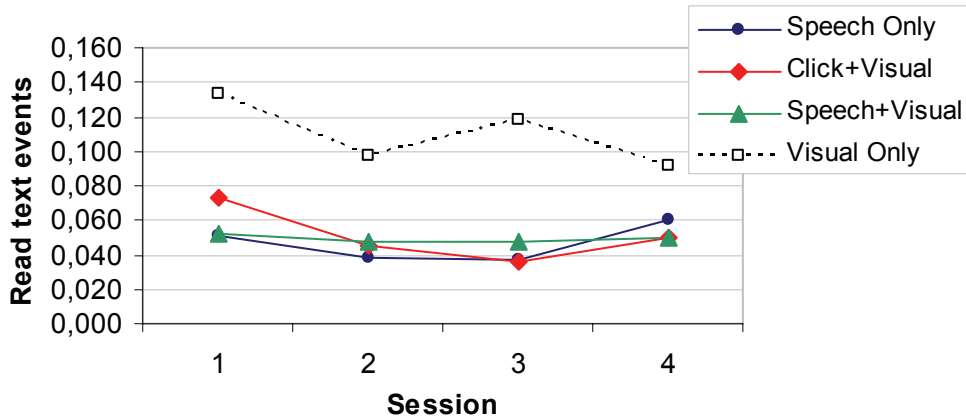


Figure 8.7. Mean RTE by feedback mode and session.

Subjective Satisfaction

The Click+Visual feedback mode was preferred by 62% of participants in the first experiment (15% Speech+Visual, 15% Speech Only, 8% Visual Only). Participants felt that spoken feedback or the “click” sound suitably supported visual feedback. The synthesized voice annoyed some participants, though.

By the end of the experiment (after four sessions of eye typing), all participants agreed that the dwell time (900 ms) was too long, even if it was appropriate at the beginning of the experiment. Participants reported that the long dwell was tiring to the eyes and made it hard to concentrate.

8.4 EFFECTS OF ANIMATED FEEDBACK

The second experiment (Majaranta et al., 2003b) was similar to the first except in its closer investigation of the “shrinking letter” condition. It was felt that shrinking not only serves as a good indicator of dwell time progress but also draws the user’s attention, thus helping the user to concentrate on the center of the key.

Participants and Design







Twenty university students (nine females, 11 males, mean age of 27) volunteered for the experiment. All were able-bodied and had normal or corrected-to-normal vision. None had previous experience with eye typing.

The following feedback modes were tested (Table 8.2):

Shrinking. This is the same as “Click+Visual” in experiment 1. Here, it is called “Shrinking” because the experiment is constrained to study only the effect of the shrinking letter.

No Shrink. The same as Shrinking, but the symbol does not shrink.

Table 8.2: Feedback modes in the second experiment

Feedback mode	While in focus	When selected
Shrinking 	highlight, shrinking 	red letter, key down, “click” 
No Shrink 	highlight 	red letter, key down, “click” 

The experiment was a repeated measures design with two feedback modes. The order of the feedback modes was counterbalanced. The results are based on a total of 200 phrases (20 participants x 2 feedback modes x 5 phrases).

Results

Typing Speed

The grand mean typing speed was 6.83 wpm. The feedback mode had a significant effect on text entry speed ($t = 2.94$, $df = 19$, $p < .01$); a significantly higher text entry rate was observed in the Shrinking mode, with a mean of 7.02 wpm, as compared to the No Shrink mode, with a 6.65 wpm mean. An explanation for the lower speed in No Shrink mode was found in the course of studying gaze behavior (discussed below).

Accuracy

In this experiment, the feedback mode did not have a significant effect on error rates or KSPC. The character level error rates were quite low (0.43%), and the grand mean KSPC value was 1.09.

Gaze Behavior

There were no significant effects of feedback mode on read text events, which measures a participant’s gaze behavior within the typed text field. However, there were significant effects on the gaze behavior within a key (on the virtual keyboard).

Re-focus events is a measure of the average number of times a participant re-focuses on a key to select it. The RFE values were studied in only the second experiment, in order to understand the effects of the shrinking letter on participants' gaze behavior within a key. Indeed, as shown in Figure 8.8, RFE was about 59% higher for the No Shrink condition (0.297) than for the Shrinking condition (0.187) ($t = 4.56$, $df = 19$, $p < .001$). The higher RFE for No Shrink indicates that participants gazed away from a key too early, before it was selected, necessitating re-focus. Therefore, the shrinking evidently helped participants maintain their focus on the key. The higher RFE probably also explains the decrease in typing speed (reported above), since re-focusing on a key takes time.

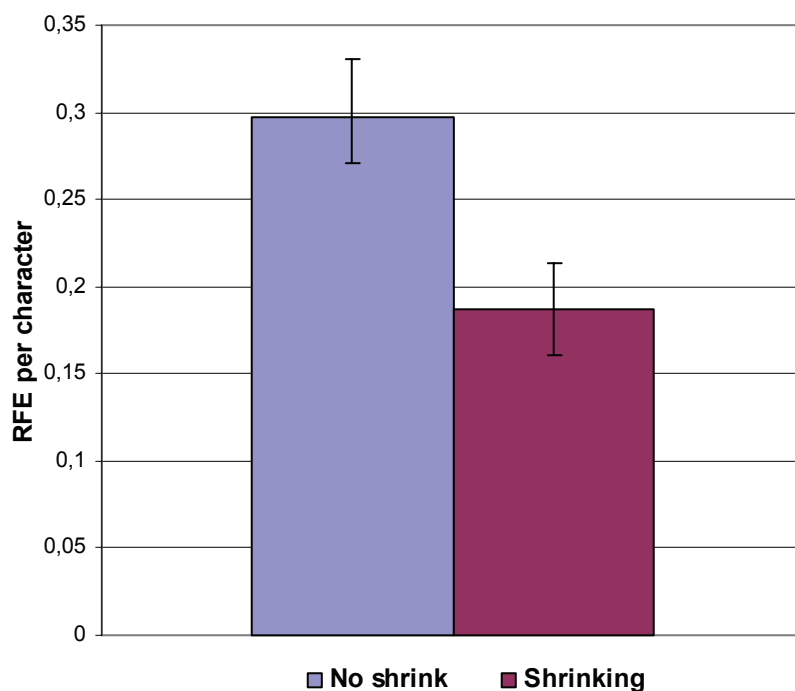


Figure 8.8. Mean RFE per character (and the standard error of the mean, SEM).

Subjective Satisfaction

50% of the participants preferred the Shrinking mode, with 65% finding shrinking to be an aid in concentrating on the key. Participants agreed that shrinking helped them understand the progression of dwell time. Some participants emphasized that “shrinking supports the typing rhythm.”

Two participants considered shrinking disturbing. Interestingly, some participants did not notice the difference between the modes, so the shrinking obviously did not disturb them. However, most participants agreed that shrinking might be disturbing and tiring in the long run, even though it helps novices to learn eye typing.

8.5 EFFECTS OF FEEDBACK WITH A SHORT DWELL TIME

In another experiment (Majaranta et al., 2004), the effects of feedback when a short dwell time is used were studied. It was felt that the results from the first experiment may not apply with short dwell times. For example, with longer dwell times, two-level feedback (focus + selection) is beneficial because the user has a comfortable opportunity to cancel before selection. With shorter dwell times, this may not be possible or may be more error-prone.

Participants and Design

Eighteen students volunteered for the experiment. On account of technical problems, data from three participants were discarded. In the end, there were 10 males, and five females (mean age: 25 years). All had normal or corrected-to-normal vision, and all had participated in either experiment 1 or 2. The experiment involved experienced participants because a shorter dwell time was used, and it was important to compare the results with those from earlier experiments.







The visual feedback was simplified on the basis of pilot tests and experiences from the previous experiments. The following feedback modes were tested (Table 8.3):

Speech Only. The symbol on the key is spoken on selection.

One-Level Visual. The key background turns red on selection.

Two-Level Visual. The key is highlighted on focus. On selection, the key background turns red.

Table 8.3: Feedback modes in the third experiment

Feedback mode	While in focus	When selected
Speech Only	none	letter spoken
One-Level Visual 	none 	red background 
Two-Level Visual 	highlight 	red background 

The dwell time for selection was 450 ms – i.e., half of that used for experiments 1 and 2 (900 ms). For the Two-Level Visual mode, the delay before highlight was 150 ms. Thus, the highlighting (before selection) lasted 300 ms. On the basis of experiences from pilot studies, the dwell

time to reselect the current letter was increased by 120 ms to avoid erroneous double entries (e.g., “aa”). Thus, the dwell time for the second of the two consecutive letters was $450 + 120 = 570$ ms.

The experiment was a repeated measures design using a counterbalanced order of presentation. The results are based on a total of 450 phrases (15 participants \times 3 feedback modes \times 10 phrases).

Results

Typing Speed

The grand mean typing speed value was 9.89 wpm. In comparison to the previous experiments, a faster entry speed was expected, since the dwell time was smaller and the participants were experienced. The feedback mode had a significant effect on text entry speed ($F_{2,28} = 6.54, p < .01$). The Speech Only mode was significantly slower than either of the two visual feedback modes (see Figure 8.9).

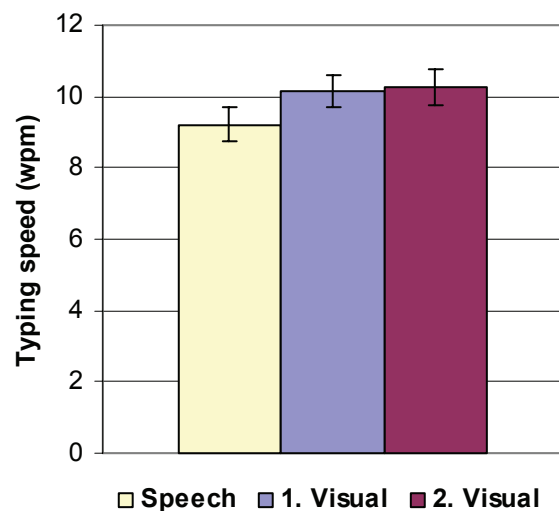


Figure 8.9. Typing speed in wpm (and SEM).

Pairwise t-tests showed that the difference in text entry speed between Speech and One-Level Visual was significant, $t = 2.72, df = 14, p < .05$. Similarly, the difference between Speech and Two-Level Visual was significant: $t = 2.87, df = 14, p < .05$. The difference between One-Level Visual and Two-Level Visual was not significant. The Speech mode was significantly slower than either of the two visual feedback modes, with a mean of 9.22 wpm. The means for the visual feedback modes were 10.17 wpm (One-Level Visual) and 10.27 wpm (Two-Level Visual).

One possible reason for the slower typing speed with Speech mode is revealed by inspecting the gaze paths. The participants spent time

listening to the speech synthesizer speaking the letter and thus did not leave the key as soon as they could have (the key was selected as soon as the dwell time had elapsed, and the dwell time for the next key started running instantly after the previous selection). By studying the audio (.wav) file recorded from the speech synthesis, we found that it took typically at least 200 ms for the speech synthesizer to speak the letter (e.g., ~200 ms for “a” and ~350 ms for “m,” with soft fading at the end). Compared to the short (70 ms) red flash that was used for selection in the visual feedback modes, the spoken feedback was quite long.

By translating the wpm measure back into search + dwell times, we get the average time spent to type a character in each feedback mode: 1300 ms for Speech, 1180 ms for One-Level Visual, and 1170 ms for Two-Level Visual. The difference between Speech and One-Level Visual is 120 ms, and the difference between Speech and Two-Level Visual is 130 ms. Since the difference is less than the time required for spoken feedback (typically more than 200 ms), one can see that the participants left the key before the spoken feedback ended. Nevertheless, the spoken feedback consumed more time. The extra time spent in listening to the spoken feedback also caused a decrease in accuracy, as discussed below.

Accuracy

The overall error rates were higher in the third experiment, but still quite low, with a grand mean of 1.20%. The increased error rate is no surprise, since there always is a tradeoff between speed and accuracy in text entry tasks. In other words, reducing the dwell time tends to push entry speed up while reducing accuracy.

The effects of feedback mode on error rate were not significant. However, the differences in KSPC across feedback modes were significant ($F_{2,28} = 9.83, p < .005$). KSPC for Speech Only (mean: 1.28) was significantly higher than the One-Level (mean 1.17) and Two-Level Visual (mean of 1.19) modes (in both, $p < .05$; see Figure 8.10). So, despite the relatively low error rates (about 1.20%), quite a few errors were committed and corrected, particularly with the Speech Only feedback mode (with 28% keystroke overhead, roughly corresponding to a 14% corrected error rate).

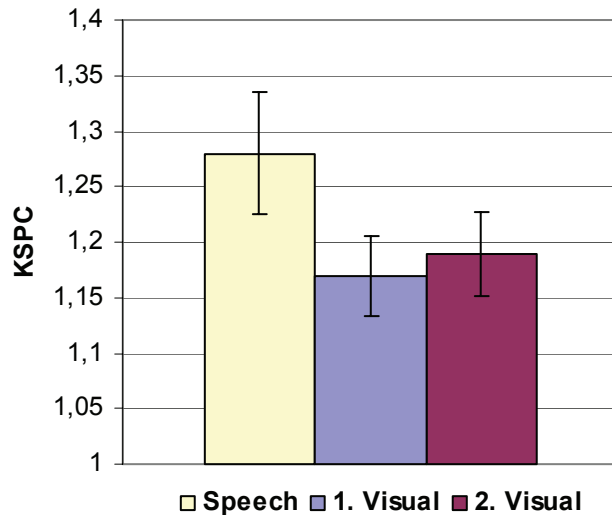


Figure 8.10. KSPC (and SEM).

As with entry speed, the higher KSPC for Speech Only in the third study was likely due to participants' tendency to pause and listen to the speech synthesizer. If the participant spent more time on the key than the specified dwell time, this caused an unintended double entry, as confirmed by a closer examination of the error types (see Figure 8.11). There were significantly more (corrected) double-entry errors in Speech Only mode than in the other two modes ($F_{2,28} = 19.12, p < .001$). Other kinds of errors included, for example, the user leaving the key before it was selected (missing character) and the user typing a wrong character (substitution).

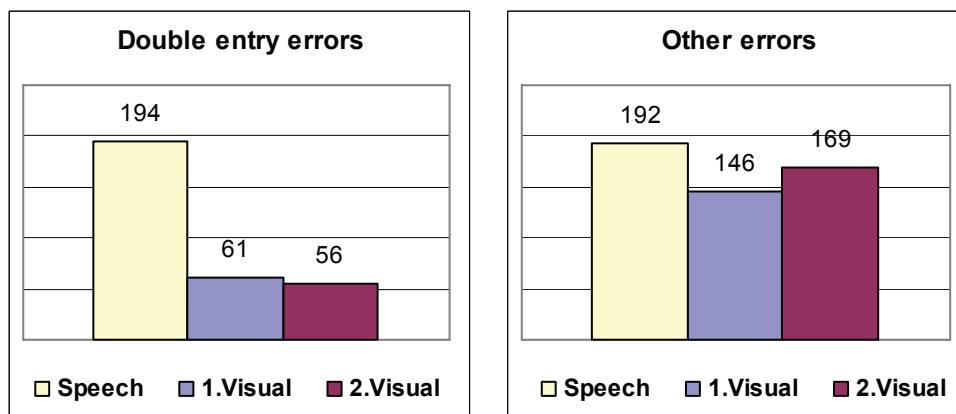


Figure 8.11. Double-entry errors (left) and other errors.

Even though the double-entry problem had been anticipated from the pilot tests, the results obtained indicate that the increase in dwell time ($450 + 120 = 570$ ms) is insufficient to avoid this for many participants, especially with Speech Only mode. One way to avoid this problem altogether is to require the user to glance away from the currently selected

key before the dwell time counter for that key starts to run. This would enable the user to stay (rest) on the key for as long as he or she desires without fear of a double-entry error or any other unwanted action.

Gaze Behavior

The feedback mode had a significant effect ($F_{2,28} = 4.50, p < .05$) on read text events, with means of 0.139 (Speech Only), 0.087 (One-Level Visual), and 0.140 (Two-Level Visual). The One-Level Visual mode had significantly fewer read text events than Two-Level Visual did ($t = 2.92, df = 14, p < .05$). Because of the greater variation in the Speech mode, the difference between Speech and One-Level Visual was not significant (after Bonferroni correction, $t = 2.70, df = 14, p < .1$). However, there was a trend towards Speech having more read text events (see Figure 8.12).

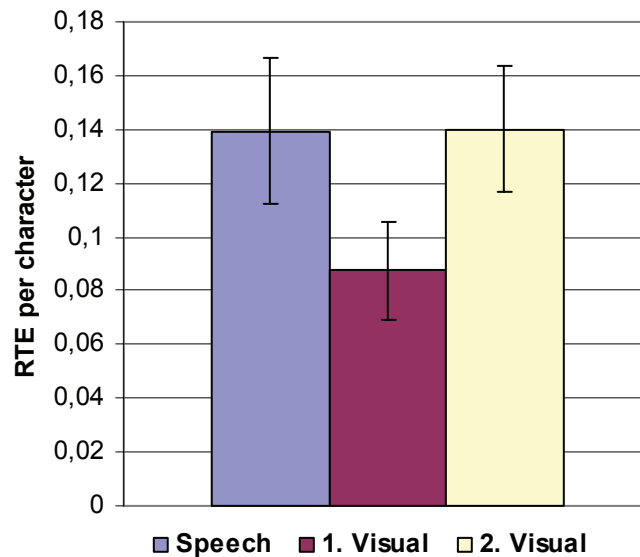


Figure 8.12. Number of read text events per character (and SEM).

The increased need to review the typed text that is seen in Speech mode is explained by the need to correct more errors. When correcting errors, the user deleted the last character (pressed Del), briefly glanced at the typed text field to see whether the deletion was successful, typed the right letter, and then reviewed the typed text again.

The increased need to review the typed text in the Two-Level Visual mode can be attributed to a degree of confusion in separating focus and selection. Participants were not sure whether the key was already selected. This was confirmed by the comments received (discussed below). With Speech Only mode, the higher RTE followed from an increased need to verify corrections (participants typically reviewed the typed text every time they corrected an error).

Subjective Satisfaction

About 47% of the participants preferred the One-Level Visual feedback, 33% liked the Two-Level Visual feedback best, and 20% preferred Speech. We asked the participants to cite reasons for their preference. The participants appreciated the simplicity of the One-Level Visual feedback. Some also felt that the One-Level Visual feedback was the most fluent and pleasant mode to type with.

Participants who preferred the Two-Level Visual feedback appreciated the extra confidence given by the highlighting: "I instantly see what letter is going to be selected and can quickly adjust my gaze if necessary." They also found the short time between focus and selection ($450 - 150 = 300$ ms) long enough to react, and to adjust the point of gaze.

Participants who liked the Speech feedback best wanted to hear what they write. They said it helped them to follow the typing and also aided in correcting errors. Many of the participants who otherwise preferred visual feedback commented that the spoken feedback helped in correcting errors. Many commented that with a slightly longer dwell time they might like the speech combined with visual feedback.

About half of the participants found the highlighting for focus in Two-Level Visual mode distracting and disturbing. They felt that it caused extra visual noise that made it hard to concentrate on the typing. The focus "flashing" around the screen also left participants concerned that something might be selected accidentally.

A third of the participants thought that the extra highlighting in Two-Level Visual did not give them any advantage over the One-Level Visual feedback. For them, the 300 ms between focus and selection was not long enough to adjust the focus of the gaze.

Twelve participants (80%) found the typing speed "just right." No one felt that the speed was too slow, but for three participants the speed was too fast. This only applies to the preferred choice of feedback. There were differences between the feedback modes in how participants perceived the typing speed. Five participants commented that the typing felt too fast in the Two-Level Visual mode, much faster than in the One-Level Visual mode (as mentioned earlier, the dwell time for selection was the same for all modes). The highlight probably caused extra stress, as one participant noted. Similarly, many commented that the speech felt too fast and for that reason caused a lot of errors (eight participants preferred the mode that was, in fact, their fastest mode).

In addition to answering our questions, participants also gave other opinions. The location of the Del key and the red color used in the visual feedback modes were noted by several participants. The location of the

Del key was not good. In the setup, the Del key was in the top right corner of the virtual keyboard (similar to the location of the Backspace key on standard keyboards). Participants commented that the Del key should be placed as near as possible to the typed text field, since they often need to check the text during or after deletion of characters. Our observations during the experiment also highlight the need to somehow help the users with correction of text: typically, the participants checked the typed text field after every correction made. This is a subject meriting further study (text editing by gaze is discussed further in Section 10.1).

The red color indicating the selection annoyed some participants. They commented that a red color is too strong in comparison to the light gray background. One participant also commented that red means denial or warning and that, thus, a more neutral color would be better. The possibility to change the color is not only important because of the user's preferences; it may actually affect performance (Wolfson & Case, 2000).

8.6 DISCUSSION

Compared to the first experiment, the typing speed in the third study was faster in all modes (obviously, since a shorter dwell time was used), and the overall error rates were higher. The decrease in accuracy is no surprise, since there always is a tradeoff between speed and accuracy in text entry tasks. However, with the spoken feedback the accuracy decreased considerably more than with the visual modes. The duration of the spoken feedback is a problem when short dwell times are used in eye typing. However, the problems with the spoken feedback in the third study do not mean that it could not work under different conditions. The length of the spoken feedback did not cause any problems with longer dwell time durations in the first study. On the contrary, Speech Only (with no visual feedback) or combined Visual+Speech produced fewer errors than did visual feedback alone. Therefore, we assume that better results could be achieved by adjusting the properties of the speech synthesis. The reason for not doing that in the third experiment is simply that we realized how many problems it caused only after the data from this experiment had been analyzed. One possibility for adjusting the speech would be to make the speech synthesizer speak more quickly and sharply, with no soft fading.

In addition to the length of the spoken feedback, speech has the problem that it cannot achieve the sharpness and clarity of the very short red flash used in the visual feedback. Thus, it may not be clear to the users whether the selection is made as soon as the speech synthesizer activates or is made only after the letter is spoken.

Methods that work well with a conventional mouse may require especially careful design with an eye mouse. The problem with the exact moment of the selection also arose from the programming point of view when the feedback of a standard “button click” event was used as feedback as such without any modification (during practice). A click consists of two events: key down and key up. The click event does not occur if the focus is moved away while the key is held down. That causes annoying errors if the user has “clicked” the key with the gaze but the selection is canceled because the gaze moves away too soon (in the very short time of showing the key visually being depressed). This error type was avoided by using the key down event as a trigger. In any event, this emphasizes the need to define a distinct point in time at which selection occurs, and to make sure the user behavior is in accordance with it.

Typing rhythm is another issue worth considering. Typing is a series of actions, including the search for the key and the selection thereof. It takes time both to search for the next key and also for the dwell time to elapse. When the same key is double-clicked, the typing rhythm is broken because the search time is not included. This was noted by a couple of participants in the third study. They would have wanted an even longer dwell time for the key repeat than that used in the experiment (the normal 450 ms plus the added 120 ms). Furthermore, a short “click” sound would have better supported the typing rhythm than visual feedback alone did. Non-speech auditory feedback has been found to support temporal tasks (with rhythm) better than visual feedback alone (Brewster et al., 1996).

The feedback itself can also affect the typing rhythm. In Speech mode, the duration of the spoken feedback varied from about 200 ms to 350 ms, depending on the letter spoken (e.g., “a” takes considerably less time to speak than “m”). It might be worth trying to normalize these times by adjusting the spoken feedback such that the durations are equal. As discussed above, the point of selection should be clear and distinct. Selection should occur immediately after the specified dwell time has elapsed, allowing the user to proceed instantly. In other words, the user should not be forced to wait for the feedback to finish.

Whether the duration of the dwell time should be adaptive (not constant for every character) is yet another interesting question. We added 120 ms for the dwell time if the user continued focusing on the selected key, to prevent false double entries. The space key might be another special case to consider. In our experiments, we have seen that novices in particular tend to either forget the space altogether or glance only briefly at the space and proceed to the next word. One explanation could be that people think in words and type words – spaces are something extra. Perhaps the dwell time for the space key should be shorter. However, if an adaptive (automatically adjusted) dwell time is used, the adaptation should not

interfere with the typing rhythm. Simpson and Koester (1999) studied adaptive scanning in an alternative communication system. They discovered that the automatic adaptation actually increased errors, because the users had developed a scanning rhythm, which the automatic adaptation interfered with (the adjustment of dwell time duration is discussed further in Section 9.3).

The double-entry errors caused by the user's gaze remaining on the same key for too long can be avoided altogether by forcing the user to gaze away from the key before it can be reselected. This gives the user all the time he or she needs to listen to the spoken feedback and to plan for the next move. This solution also supports personal typing rhythm: the user can define the pace for double entries. This kind of solution used by some of the assistive keyboards. From personal observation, I know that users learn quite effortlessly to glance away and back to the key to reselect it in order to make double entries.

Most of the participants appreciated the "click" in the first study, and participants in the third expressed a desire to have it in that study also. They commented that the very short red flash did not seem to be enough. The "click" was left out of the third study to simplify the experimental setup. However, we assume that a "click" sound would have helped to improve typing performance.

Many participants felt that visual feedback is very important and that spoken feedback alone is not sufficient. As a couple of participants commented, it was sometimes hard to discriminate some letters from the spoken feedback alone. For example, "n" and "m" sound quite similar, and they are also located next to each other in the QWERTY keyboard layout. Added visual feedback would have confirmed the selection.

We were slightly surprised that the Two-Level Visual feedback in the third study did not cause more problems and increase error rates. As demonstrated by this experiment, for many users, 300 ms is a long enough time to react (by gaze); participants actually corrected their point of gaze in the short time interval between the focus (at 150 ms) and the selection (at 450 ms). Thus, dwell times as short as 300 ms are possible. As commented by participants who had participated in both the third experiment and one of the experiments with a longer dwell time, "faster is better."

Animation was found to be useful for long dwell times in the second experiment. It helped the users to maintain focus on the button long enough for the dwell time to elapse and the button to be selected. However, the users should be able to adjust the dwell time soon after they learn the basics, to avoid frustration caused by the unnecessarily long dwell time duration.

We used a shrinking letter as animated feedback to indicate the progression of the dwell time. The shrinking letter approach worked well for our experiment. However, as a general approach it may be impractical to shrink the target itself. If the screen button contains a command instead of a single letter, shrinking would make it hard to read the button label (the command name). Therefore, showing a transparent shrinking dot or a closing circle (see the feedback on the letter “i” in Figure 9.11) on the key would provide a more general, practical animation.

8.7 GUIDELINES

When gaze is used for both input and output, conventional guidelines for graphical user interfaces may not be suitable as such; in such general guidelines, it is assumed that the gaze is free for observation and its special characteristics as an input method are not considered.

Stemming from the results of the experiments, as well as knowledge from previous research, the following six guidelines on feedback in eye typing were formulated.

1. *Use a short non-speech sound to confirm selection.*

Visual feedback combined with a short audible “click” produced the best results in the first experiment and was also preferred by the participants. Even though “click” sounds were not tested with a short dwell time, many participants in the third experiment commented that the visual feedback would benefit from an additional “click.” This is consistent with previous research. A non-speech sound not only confirms selection but also supports the typing rhythm better than visual feedback alone (Brewster et al., 1996; Brewster & Crease, 1999).

2. *Combine speech with visual feedback.*

Even though Speech Only mode produced good results in the first experiment (with a long dwell time), it was not liked by participants. Some of the participants found the spoken feedback on every keystroke quite disturbing. Furthermore, some letters are hard to distinguish by speech alone. Spoken feedback is especially problematic with short dwell times, since speaking a letter takes time. As demonstrated in the third experiment, people paused to listen to the speech. This, in turn, not only decreased typing speed but also decreased accuracy. Since the time to speak a letter varies (e.g., “e” v. “m” v. “w”), spoken feedback does not support the typing rhythm (especially with short dwell times). However, if speech is combined with visual feedback, it can improve performance, as seen in the first experiment, and may be helpful, especially for novices.

3. Use simple, one-level feedback with short dwell times.

Short dwell times require sharp, clear feedback; it should be temporally and visually precise. For example, spoken feedback may be problematic, because it may not be clear to the user whether the selection occurs at the beginning or end of the synthetically spoken letter. When a short dwell time is used, there is no time to give extra feedback to the user. As seen in the third experiment, the two-level feedback was confusing and distracting, even though the measured performance was reasonably good. With very short dwell times, the feedback for focus and selection are not distinguishable anymore. The feedback should have a distinct point for selection; there should be no uncertainty of the exact moment when the selection is completed.

General usability guidelines (e.g., Nielsen & Mack, 1994) indicate that feedback on actions should be provided within reasonable time. This also applies for gaze input. As the results of the experiments conducted show, people may gaze away from a key too early if feedback is delayed. Thus, a simple one-level feedback system may not work well with long dwell times. A possible solution is to give feedback for focus before selection occurs. The separated two-level feedback (focus + selection) should, however, be designed carefully, to avoid confusion.

4. Make sure focus and selection are distinguishable in two-level feedback.

As seen in the experiments, two-level feedback may cause confusion. A simple audible click helps to make the moment of selection distinct and clear. As seen in the second experiment, using a click in both conditions, the shift between focus and selection is also strengthened by animation.

5. Use animation to support focus with long dwell times.

Since animation takes time, it is difficult to use with short dwell times, or users may (at least at first) find it confusing (Hansen et al., 2003a). However, with long dwell times, animation provides extra information on the dwell time progress. It is not natural to fixate for a long time on a static target (Stampe & Reingold, 1995), so animation helps in maintaining focus on the target letter for long dwell times. In the second experiment, the shrinking letter improved performance by helping users focus on the center of the key (though shrinking the letter itself may introduce problems as the letter gets smaller and harder to see). Animation gives continuous feedback for continuous waiting (for the dwell time to end). The animation should be designed carefully; it should be subtle and not distract the user from the task at hand. To the greatest extent possible, animation should show - in a direct, continuous fashion - the time remaining to selection.

6. *Provide the capability to adjust feedback parameters.*

The dwell time should, of course, be adjustable. The same 500 ms may be “short” for one user and “long” for another. The needs and the preferences of users vary a great deal; this is especially true for people with disabilities (Hutchinson et al., 1989; Donegan et al., 2005, 2006a, 2006b). Therefore, the final guideline is to support user control of feedback parameters and attributes.

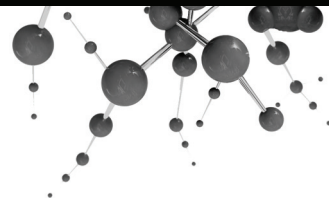
It should be noted that none of the studies reported upon above show the eye cursor. Therefore, one should be aware that showing a cursor may considerably affect the results and guidelines presented here. For example, a cursor that moves along the gaze path always gives immediate feedback on the focus and may make the extra feedback on focus unnecessary. This is a topic worth exploring in further studies.

8.8 CONCLUSION

The results of the experiments indicate that the type of feedback has a significantly impact on typing speed, accuracy, gaze behavior, and users’ subjective experience. Furthermore, dwell time duration affects the suitability of certain types of feedback. Even though the differences were not especially large in some cases (a slight increase in accuracy or a few extra characters written as a result of increased speed), they are important in a repetitive task where the effect accumulates. Users may adapt to the shortcomings of the feedback up to a point. However, as seen in the first experiment, the effects (on performance and accuracy) were still significant after four sessions. Nevertheless, since the participants in the experiments were either first-time users (experiments 1 and 2) or had only a little experience (experiment 3), the results apply best to novices.

Naturally, participants’ preferences varied in all experiments, but some consistent opinions were found. For example, the use of an audible “click” was generally liked. Participants also appreciated feedback that clearly indicates selection, as well as feedback in support of their typing rhythm.

Typing rhythm is considered important because dwell time, as an activation command, imparts a sense of rhythm to the task. When typing “as fast as possible,” the participant no longer waits for the feedback but learns to take advantage of the rhythm inherent in the dwell time duration. In other words, the interaction (type a letter, proceed to the next letter) is no longer based on reaction time but follows from the rhythm imparted by the dwell (and search) time. Rhythm-based eye typing may also exacerbate the problem with erroneous double entries, since the typing rhythm is interrupted on account of the reduced search time.



9 Learning to Write by Gaze

9.1 LEARNING VOLUNTARY GAZE CONTROL

The eye is a perceptual organ, not evolved for control. Even though it is fairly easy and natural to point at items by gaze, it takes some time to learn to use gaze as a means for controlling a computer. Bates (2002) compared novices (with less than two hours of practice with gaze control) and slightly more experienced users (with more than six hours of practice) and observed noticeable differences in their gaze behavior. Inexperienced users reviewed the text they had written so far more often, they had more problems with pointing accuracy, and they also made more errors caused by inadvertent dwell clicks when compared to slightly experienced users.

Learnability is one of the attributes of an interface with good usability (Nielsen & Mack, 1994). In designing new interaction techniques and novel interfaces for non-experts and people with special needs, it is important to ensure the usability, and learnability, of the system. Aoki et al. (2008) studied the characteristics of learning to interact with a computer by gaze. They measured the effect of learning gaze control by measuring various gaze-related actions and how they develop over time. They developed a taxonomy of basic gaze actions involving dwell-time-activated keys. They first classified the gazes into two groups: gazes that activated a key, meaning gazes where the dwell time duration exceeded the predefined dwell time threshold (500 ms) and gazes with dwell time durations below the threshold, indicating that the user only had a brief look at the key but did not select it (a typical threshold for the minimum fixation is 100 ms – see, e.g., Jacob & Karn, 2003). Gazes that selected a key were then categorized further, into correct key activations and erroneous

activations. Learning to select a key by dwell time (during gaze typing) was reflected in these three measures: number of gazes for 1) correct selections, 2) incorrect selections, and 3) attended keys not selected (AKNS). Results from an experiment with eight participants and 22 experimental sessions (taking a total of roughly 10.5 hours, on average) showed significant learning effects according to all three metrics. The number of erroneous key activations, as well as the number of keys that were attended but not selected, decreased rapidly during the first three sessions. Also the number of correct key activations per character typed became stable within the first two or three training sessions, indicating that most learning happened during the first sessions.

Aoki et al. (2008) also compared the three measures of dwell-time-based gaze activations with general measures of typing performance, such as characters entered per minute (cpm)¹⁹. They found that control of eye movements (dwell-time-based gaze activations) is gained more quickly than the typing speed increases. They interpret this as indicating that voluntary gaze control can be learned easily. According to them, the common belief that input by gaze is hard to learn stems from the analysis of typing performance; typing speed tends to be poor especially in the early stages, but this does not necessarily mean that gaze control is hard *per se*.

As discussed earlier (in Chapter 8), proper feedback may facilitate the learning process. For example, novice users may benefit from animated feedback that shows how the dwell time is progressing (Majaranta et al., 2003b). The animation can reduce the number of re-focus events, where the user first (correctly) dwells on the key but leaves it too early, before it is selected, and therefore is required to return to it to select. Obviously, not all brief gazes indicate that the user left the key too early; in the beginning, the user may need to fixate on several keys while searching for the correct key. This is reflected in the AKNS metric of Aoki et al. (2008), described above. Learning the interface and layout of the keyboard takes some time, depending on the level of familiarity with the layout. For example, an alphabetical layout may be faster to learn if one has no previous experience with QWERTY. The learning time is even longer for interfaces where the layout changes, such as in GazeTalk, which shows only a few of the most probable letters at a time, with the layout changing after every selection (Hansen et al., 2003b). This is reflected in longer search times and

¹⁹ Characters per minute (cpm) was used instead of words per minute (wpm) because Aoki et al. (2008) experimented with a Japanese version of the GazeTalk application and wpm is not suitable for the Japanese typing system.

in the AKNS measure, since the user needs to first locate the item (the letter on the keyboard) before it can be selected. Obviously, if one wishes to learn a totally new writing technique such as a gesture-based alphabet, it takes time to master it (see, e.g., Wobbrock et al., 2008).

Joos et al. (2007) compared two different gaze typing systems, GazeTalk (Hansen et al., 2001) and the Eyegaze system by LC Technologies (Cleveland, 1994). GazeTalk provides a 4 x 3 grid with large keys that are easy to hit. It has a hierarchical layout and dynamic character and word prediction features (for a general introduction to GazeTalk, see Section 5.2; see also sections 6.2 and 6.3 for a description of the principles of GazeTalk's prediction features). The Eyegaze system is essentially an on-screen keyboard with a flat, static layout.²⁰ Dwell selection duration was set to 800 ms for both systems. Results from a within-subjects study with four locked-in participants suffering from ALS and five sessions showed better immediate usability and learnability for the Eyegaze system. The typing speed was significantly higher for Eyegaze, and the difference even increased in time. In terms of the task efficiency (which takes into account error rate and task completion time), Eyegaze was significantly better than GazeTalk, with an efficiency value approximately three times higher in the last session. Participants' subjective ratings were in line with the performance measurements; the Eyegaze system was judged more positive on all scales. When interpreting these results, one should keep in mind that GazeTalk was developed for trackers with low spatial resolution. Its hierarchical layout means that the user has to make about three times more selections per character than in the explicitly flat design of the Eyegaze layout. Even though GazeTalk includes word and character prediction, which can increase typing speed, it obviously takes more learning time to take full advantage of the prediction features.

In the following sections, we will describe two longitudinal experiments with gaze-based text entry. In both, we were interested in studying how rapidly novices can learn to enter text by gaze alone. The systems tested use totally different text entry methods: the first, Dasher, includes character prediction and uses continuous gestures, whereas the second, a dwell-time-operated (flat) on-screen keyboard, uses direct pointing and has no prediction features.

²⁰ A detailed description of the Eyegaze system is available at <http://www.eyegaze.com/>.

9.2 LEARNING TO WRITE BY GAZE VIA CONTINUOUS GESTURES

Dasher has been one of the most discussed inventions in the area of gaze-based text entry in recent years. When the Dasher article by Ward and MacKay (2002) was published in *Nature*, it attracted worldwide interest in the public press. According to the piece, Dasher was about twice as fast as any of the previous gaze writing systems, and five times more accurate. Hence, it created a lot of excitement for people working with interactive eye tracking and among people with disabilities. Dasher is freely available in more than 60 languages, and it seems to be highly appreciated by users with disabilities (see comments from users quoted on the Dasher Homepage, 2008). Despite all of the attention, no independent experiments on gaze writing with Dasher had been published to verify the results described by Ward and MacKay.

We (Tuisku et al., 2008) conducted a longitudinal study wherein 12 participants transcribed Finnish text with Dasher in ten 15-minute sessions, using a Tobii 1750 eye tracker as a pointing device. In addition to the aforementioned motivation, we were interested in how easy Dasher is to learn when one is using only an eye tracker and exactly how proficient new users can become after a few hours of practice. We also hoped to gain insight into the pros and cons of eye-controlled Dasher, for example, to find out how straining it is for the eyes (from the participants' point of view) and what the typical problems in learning and using it are.

We will start by explaining how Dasher works and why it is so well suited to gaze pointing with an eye tracking device. We will then briefly review related work before going into details of our study.

Gaze Writing with Dasher

Dasher (Ward & MacKay, 2002) is a text entry interface that is operated via continuous pointing gestures. Writing happens by zooming into a world of characters. In the initial state, all characters are in alphabetical order on the right side of the screen (see Figure 9.1).

The user writes by moving the pointer towards the desired character(s). The Dasher interface zooms in, and the area around the character pointed to starts to grow and move towards the center of the screen. As soon as the character crosses the central vertical line, it is selected and entered in the text box at the top of the screen (see Figure 9.2).

While the interface zooms in towards the focused character, the language model in Dasher predicts the most probable next characters. The areas of the most probable characters start to grow within the region of the chosen character as it moves leftward. This brings the most probable next characters closer to the current cursor position, thus minimizing the distance and time for selecting them.

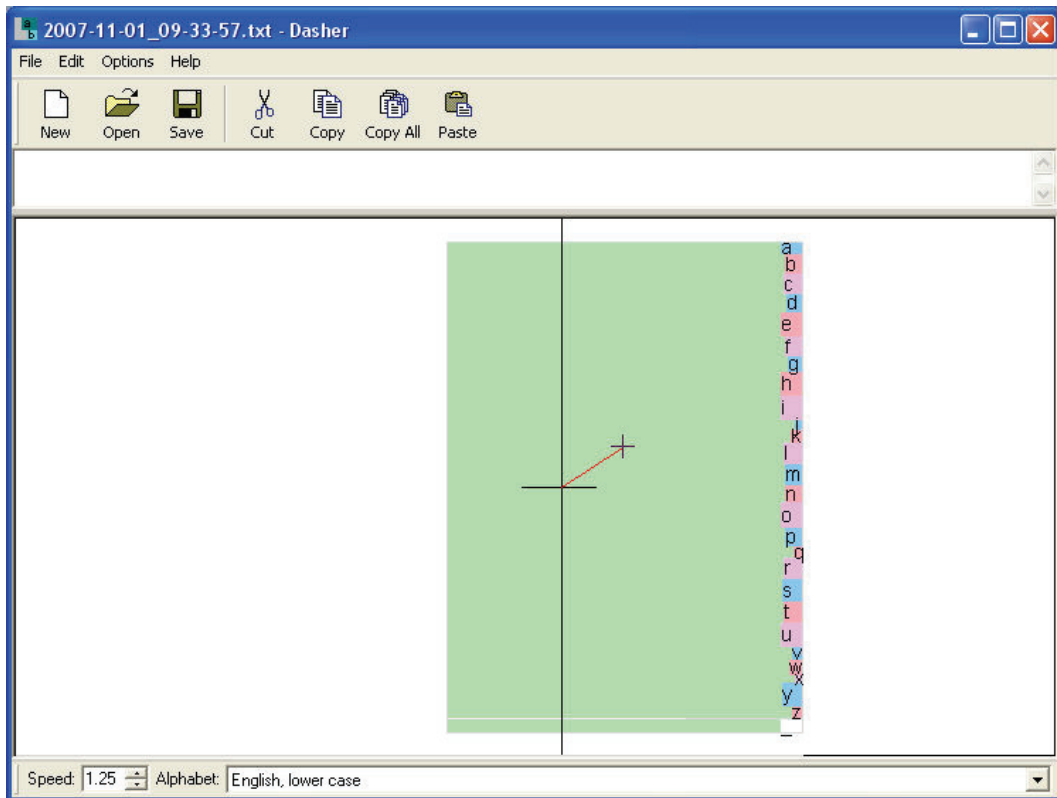


Figure 9.1. Dasher in its initial state.

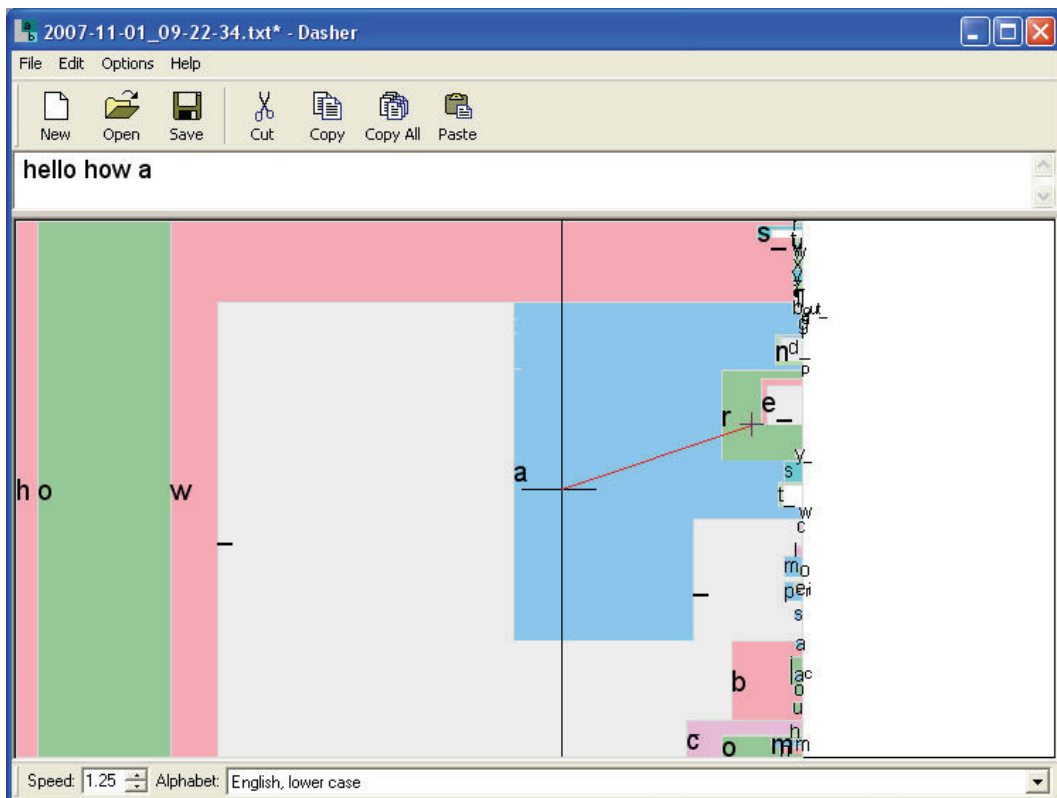


Figure 9.2. Dasher zooming in to the world of characters. The user is in the middle of writing “hello how are you,” with the letter “a” of the word “are” just written.

Canceling entered characters is done simply by pointing left, which inverts the direction of the action. Instead of zooming in, the interface zooms out and the written characters return from the left back to the right side of the screen. The central vertical line acts as a home position. All action ceases when the user holds the cursor at the center of the screen.

Dasher can be controlled with any two-dimensional pointing device, such as a joystick; stylus; trackball; or hand-, head-, or foot-operated mouse. There are also implementations of Dasher for other input devices, such as tilt sensors, breath-control, and buttons (switches) (Dasher Homepage, 2008). However, controlling Dasher via eye movements has created the most enthusiasm because Dasher seems especially suitable for eye control.

No additional switches or dwell time are needed for using Dasher with gaze. The user simply looks at the characters. An eye tracker follows the user's gaze and moves the cursor to whatever point the user is looking at. Dasher works particularly well with gaze pointing because the desired character – and the string of the following characters – is at the focus of the user's attention and, therefore, the user's eyes are naturally pointing at it.²¹

Binding the cursor to eye movements has its disadvantages, which are present in Dasher, too. Using a mouse or any manual pointer, the user is able to look around while continuing to point at one location. With gaze, the cursor always follows the gaze, preventing the user from looking around without moving the cursor. For example, it is difficult to review the written text without deleting parts of it; moving the gaze to the top left also moves the cursor toward the left and may initiate canceling. To prevent this from happening, it is recommended to pause Dasher before reviewing text and restart when ready.

Research has shown that the accuracy problems can be compensated for by means of fisheye lenses (Ashmore et al., 2005) or zooming (Bates & Istance, 2002). Similarly, Dasher's zooming interface is able to alleviate the accuracy problems to a certain extent. Even if the cursor does not hit the target character's area directly, it will be selected when its region grows as the interface zooms in toward the next characters within its region. If the accuracy is way off, the writing will be slowed down considerably. However, if the accuracy is reasonably good, "driving" the gaze smoothly through the characters within a relatively small area may be more comfortable to the eyes than an on-screen keyboard where the user has to

²¹ Because of Dasher's dynamic nature, gaze writing with it has sometimes been described as feeling as if one is reading the sentence to be written while the next characters and even whole words smoothly follow the characters already written, rather than as selection of individual letters.

constantly move the gaze from one side of the screen to the other to select each character, one at a time.

Other advantages of Dasher include the built-in language model and speed control. The embedded character prediction makes separate word lists unnecessary, since highly probable words will appear in the Dasher user interface automatically. This saves time and cognitive effort: the users do not need to go through a separate list to see whether the word they want to write is there. It also makes writing easier and reduces error rates, because probable strings get more space and are thus easier to write (Ward & MacKay, 2002). The zooming speed can be controlled easily through gaze position. The action slows towards the center and increases towards the sides of the screen. All action ceases in the center of the screen, offering a resting position for the gaze and time for the user to think. This is important because during writing, Dasher requires sustained visual attention from the user. Bystanders have sometimes expressed concerns about the potential strain of the constant visual “noise” to the eyes. However, this seems to be less disturbing to the person controlling the cursor position.

Related Work: Studies of Gaze Writing with Dasher

In their original eye-controlled Dasher study, Ward and MacKay (2002) report a top speed of 25 wpm. The accompanying figure shows that after an hour of practice, the speed varied between 10 to 25 wpm, depending on the user (two were novices, two experts). They compared the experts’ top speed to that with a QWERTY on-screen keyboard (WiViK with its word-completion feature enabled), in which expert users achieved 15 wpm, with error rates five times that seen with Dasher. The results are encouraging, but it should be noted that the result set is based on only four participants. The authors do not report how much practice the experts had had before the one-hour experiment. The participants reported that they felt the on-screen keyboard to be more stressful than Dasher. This was mainly because with the on-screen keyboard the users were uncertain about potential typing errors. This, in turn, prevented the word prediction program from functioning correctly. The on-screen keyboard also required extra mental effort, because the users moved their gaze from the keyboard to the word list that contained potential words predicted on the basis of the first few characters already written by the user. Thus, it may have been Dasher’s in-built language model in combination with the interface design that made it more pleasant to use.

The results with eye tracking are comparable with a previous experiment by Ward et al. (2000) with a mouse as the steering device. After 60 minutes of practice (in 12 five-minute sessions), the text entry rate varied from about 12 to 25 wpm. After a few hours of practice, one experienced

participant (one of the authors) could write up to 34 wpm, which is comparable to handwriting speed (Wiklund et al., 1987).

Itoh et al. (2006) compared Japanese gaze writing in Dasher with that in two versions of GazeTalk (Hansen et al., 2001). Overall, there was no significant difference between the systems. Both achieved a text entry rate of 22–24 Kanji characters per minute (cpm), with performance improving from 19 to 23–25 cpm over seven short trials in the space of three days. The two systems also elicited similar subjective responses. For example, neither of the systems induced motion sickness which was one of the questions asked. However, there was a noticeable difference in backspacing rate, which was significantly higher with Dasher than with the versions of GazeTalk (0.0028, 0.0029, and 0.053 backspaces per typed character for the two versions of GazeTalk and for Dasher, respectively).

Urbina and Huckauf (2007) compared three new “dwell-time-free” eye typing approaches (Iwrite, StarWrite, and pEYEdit – see Huckauf & Urbina, 2007) with Dasher and a traditional QWERTY on-screen keyboard. They only report preliminary results without statistically significant differences. They did not use prediction with any of the systems. Without prediction (i.e., with a flat probability distribution where all characters had an equal share of the screen real estate), Dasher lost all its speed advantage. While the participants were able to type 10 to 15 words per minute with the QWERTY keyboard (with 500 ms dwell time), the average speed in Dasher was only 4.7 wpm, and 7.4 wpm for the fastest writer.

The experiments described above have been conducted with able-bodied participants. People who have no prior experience of typing or voluntary control of a computer may require a long time for learning (Gips et al., 1996) and many introductory activities (Donegan & Oosthuizen, 2006) before they are able to benefit from advanced gaze writing systems such as Dasher.

Method

Participants

Twelve able-bodied university students volunteered for the experiment (five males and seven females, from 21 to 30 years of age). All were native speakers of Finnish. Eleven of the participants reported normal or corrected-to-normal vision. One person reported having poor vision, but this was not, however, noticeable in any way in the experiments. One participant had seen Dasher before, but she had not used it herself. All participants were novices in gaze writing. Two participants reported that they had previously tried an eye-controlled on-screen keyboard for about five minutes.

All participants were rewarded with four movie tickets. To maintain the participants' high motivation throughout the 10-day experiment, we informed them after the first session that the participant learning the best to use Dasher would receive an extra prize. Here, "best" was defined as with the most improvement in his or her performance during the experiment, when the initial performance measured in the first session was compared to his or her performance at the end of the experiment.

Apparatus

Dasher (version 4.4.1) was run on a personal computer with the Windows XP operating system. We used the Tobii 1750 eye tracking device with its integrated 17-inch TFT color monitor (at a resolution of 1280 × 1024 pixels) to track the user's gaze. For mouse emulation, we used the eye mouse included in MyTobii (version 2.3.1.0). In MyTobii's Mouse Settings dialog, "Speed" was set to the fastest setting (thus minimizing smoothing since it would slow down Dasher's reaction to gaze even if it did make the cursor move more jerkily) and "mouse click" was set to be off.

The Dasher alphabet "Suomalainen / Finnish with punctuation and numerals" was used. Dasher was set to "Eyetracker mode" with the "eyetracker autocalibration" option on. The "Eyetracker mode" changes the dynamics of Dasher to better suit navigation by gaze, and the "Eyetracker autocalibration" option automatically detects and corrects vertical calibration errors in the gaze tracker. We placed the Dasher window in the center of the screen so that a small margin was left above and below the Dasher canvas (the canvas size was 1025 × 640 pixels, excluding menu bars and other window elements), as suggested in the Dasher Manual (MacKay, 2006).

"Dasher Speed" was initially set to 0.21 (information rate measured in bits per second (Ward et al., 2000)) on the basis of pilot tests. Dasher Speed defines the maximum writing speed of Dasher and affects the speed of the dynamic animation (how rapidly the letters are moved from their initial position on the right to the center of the screen), thus affecting the speed with which characters are selected. We set the "Adapt speed automatically" option to be on, so that Dasher would increase its speed automatically as the participants' skills improved. With this parameter set to be on, the same algorithm was used to adjust the speed for each participant, thus eliminating the potentially subjective element involved in manual adjustments. After every session, the participant's end speed was saved, so that the participant could continue with the same speed in the next session. Similarly, each participant also had his or her own training text file with "language model adaptation" set to be on.

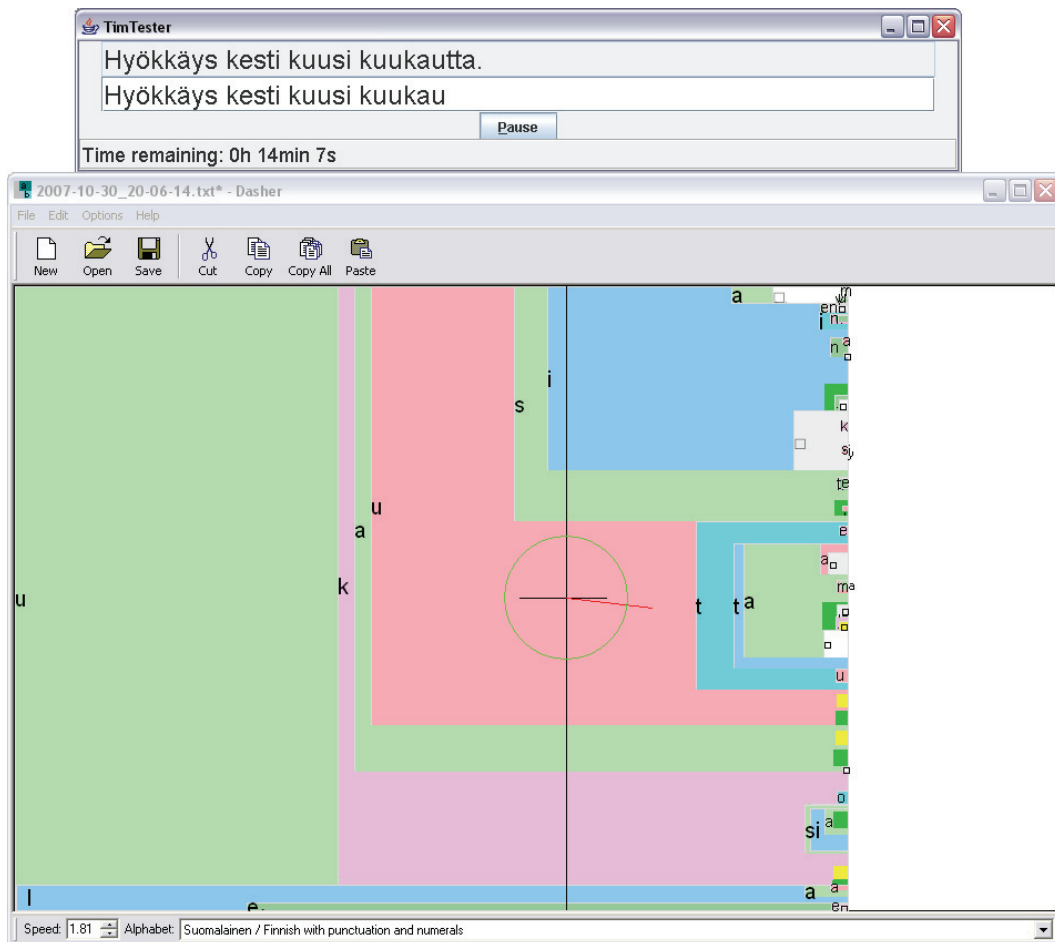


Figure 9.3. The text written by the user was directed to the experimental software that presented the target phrase.

By default, Dasher is started and stopped via a mouse click. Since we wanted to study how people learn to use Dasher through eye movements alone, we enabled an option that allows starting and stopping according to the mouse cursor position. The “Start with mouse position” option with the “Center circle” attribute was turned on in order to allow the participant to stop or start Dasher with gaze simply by looking at the circle (dwell time selection is applied inside the circle to prevent starting or stopping by accident). The circle is transparent when Dasher is on (as in Figure 9.3) and red when Dasher is stopped. We also set the attribute “Pause outside window” to be on so that every time the participant would look up to see the given phrase in the experimental software (described below) Dasher would pause.

Finally, Dasher was set to a direct-entry mode so that everything the participant wrote could be directed to the software we used to present the stimulus phrases (see Figure 9.3). We used Dasher’s log file to analyze the results.

Procedure and Design

Each participant was first briefed about the motivation for the study and eye control in general. After the briefing, Dasher was briefly introduced to the participant by the experimenter, using a mouse. The eye tracker was then calibrated and the participant was allowed to practice eye control freely with an on-screen keyboard (the one included in MyTobii) to get an idea of how typing by gaze is normally done and how slow it can be.

For the experiment, the participant sat in front of the monitor, at a distance of 50–60 cm from the monitor. Participants were instructed to sit still. However, their movements were not restricted in any way. The eye tracker was calibrated at the beginning of every session, and sometimes also during the session, if the participant expressed the need for re-calibration (when the cursor had drifted too far from the actual point of gaze, making it hard to control Dasher). If re-calibration occurred during a session, we tried to do it between phrases and the experimental software was set to “pause” during calibration. If re-calibration had to be done in the middle of a phrase, that phrase was ignored in the analysis.

The experimental task was to write as many phrases as possible with Dasher, in Finnish, within the 15-minute time limit. Participants were instructed to first memorize the phrase and then write the phrase as quickly and accurately as possible. The participants were instructed to correct errors if they detected them on the Dasher screen. If they detected errors in previous words, they were to ignore them. The phrases were from the Finnish translation by Isokoski and Linden (2004) of the 500-phrase set originally published by MacKenzie and Soukoreff (2003). The phrases were easy to remember, neutral everyday sentences. Some of them contained capital letters and punctuation; some had only lowercase letters.

The phrases were presented one by one, using Java-based software (TimTester) designed especially for text entry experiments by Isokoski and Raisamo (2004). After each phrase, the participant had to enter the Enter character (included in the Dasher alphabet) to load a new phrase. The program was set to stop when 15 minutes had passed, but if the participant was in the middle of writing a phrase, the software waited until the participant had finished the phrase before closing down. The window of the experimental program was placed above Dasher, to let the user see the phrases easily, as shown in Figure 9.3.

Each participant visited our eye tracking laboratory 10 times in June 2007. We organized the sessions such that there never was a gap of more than two days between consecutive sessions. The first and the last session took about one hour; other sessions lasted about half an hour, including preparations such as eye tracker calibration. Each participant completed 10 15-minute writing sessions by gaze and one 15-minute session with a

mouse. In total, each participant wrote for two and a half hours by gaze and 15 minutes with a mouse. The mouse session was left for the very end of the trial series, because the main goal was to study how people learn to write with eye movements alone. We were interested in comparing the results with gaze to those with the mouse, but we did not want to corrupt the gaze data by allowing the participants to control Dasher by any manual means. We chose the mouse instead of other potential control devices because it was easily available and we assumed the participants would not require extra training in using it.

Results

The results are based on data from 11 participants. One participant was a clear outlier and therefore excluded from the statistics but included in the figures (indicated with a red dashed line). Results from a few sessions are missing because of technical problems; the missing values were replaced with an average of the previous and the next session's values.

Analysis of a phrase started with entry of the first character and ended with the last character that was part of the phrase. We excluded the Enter character (which ended writing of the current phrase and loaded the next) from the analysis because it took a long time for the participants to find it. It was located at the end of the set of punctuation marks and was difficult to find. While participants were searching for Enter, they sometimes accidentally entered extra characters; these too were excluded from the analysis.

Text Entry Rate and Dasher Speed

The text entry rate (using Dasher by gaze) for each participant in the 10 sessions is shown in Figure 9.4. The grand mean writing speed was 2.49 wpm in the first session and 17.26 wpm in the tenth session; thus, one can see that significant learning occurred during the experiment. The highest session average was 23.11 wpm, reached by participant 9 in session 9.

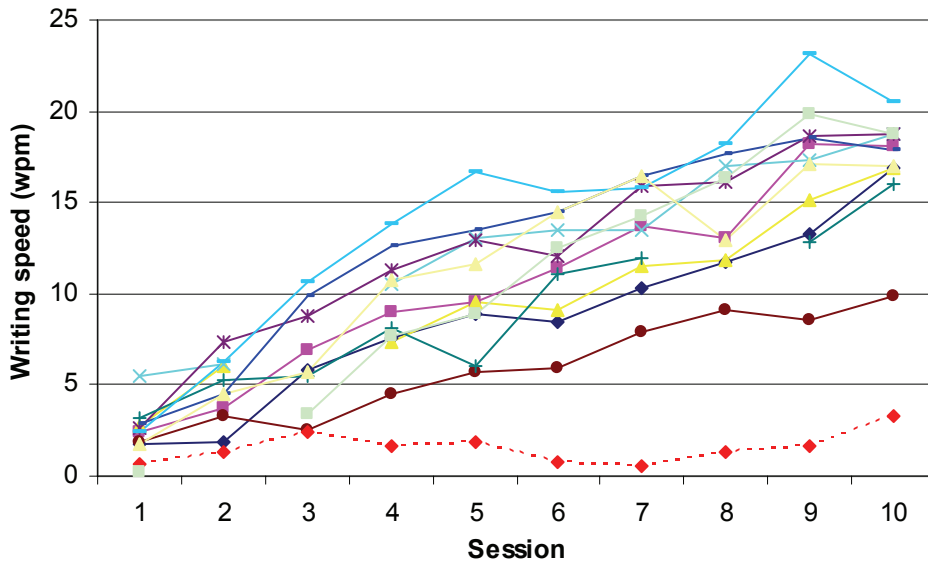


Figure 9.4. Writing speed (wpm) for each participant in the 10 sessions, using Dasher by gaze. The lowest (red dashed) line represents the outlier who never got past 5 wpm.

Dasher Speed was initially set to 0.21 bits per second for all participants. As seen in Figure 9.5, Dasher Speed increased significantly for all participants during the first four sessions. The average Dasher Speed increased to 2.15 during the first four sessions. After that, the rapid increase of the Dasher Speed figure leveled off. The grand mean was 0.76 bps in the first session and 2.63 bps for the tenth session.

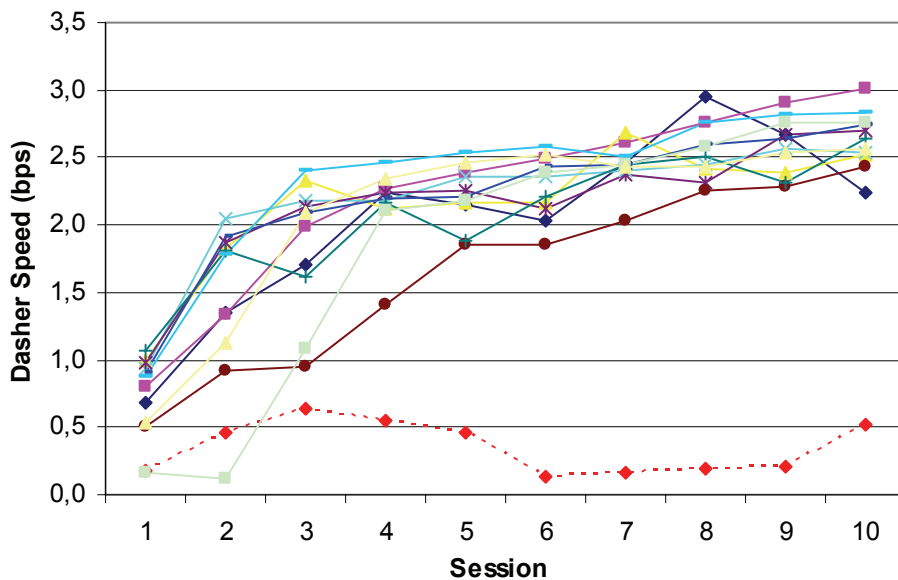


Figure 9.5. Dasher Speed (bps) for each participant in the 10 sessions. Again, the outlier (marked with a red dashed line) is far below the others.

Figures 9.4 and 9.5 are somewhat cluttered. Because of this, figures 9.6 and 9.7 show the average text entry rate and Dasher Speed only. These values were computed without the slowest, outlier participant.

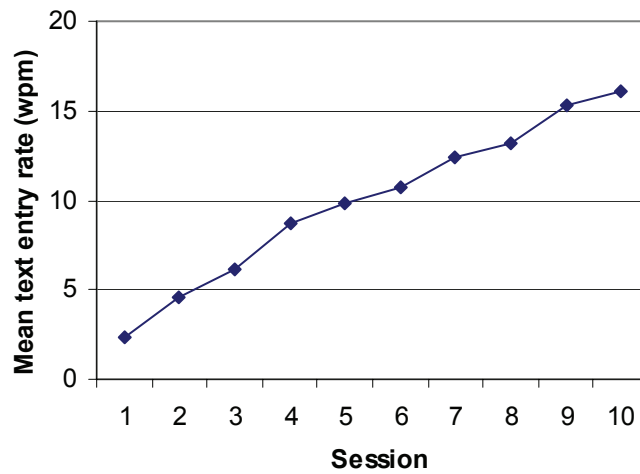


Figure 9.6. Mean text entry rate per session.

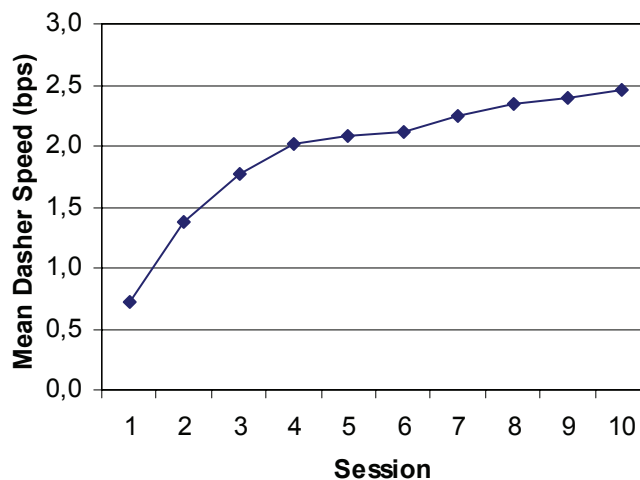


Figure 9.7. Mean Dasher Speed in bits per second.

The shape of the text entry rate v. session number curve shown in Figure 9.6 is unusual. Usually longitudinal text entry experiments result in a text entry rate curve that shows rapid growth during the first few sessions and then diminishing gains with further training (see, for example, Isokoski & Raisamo, 2004). The curve in Figure 9.6 is almost linear. Because of this, we omit the conventional fitting of a power curve to extrapolate the development of the text entry rate beyond our data. We have no confidence as to the accuracy of such prediction in this situation.

The Dasher Speed curve in Figure 9.7 shows the decelerating increase that we were expecting to see in the text entry rate curve. Our interpretation of this is that our experiment was not long enough for the participants to

reach the level where their learning rate would start to decrease. However, the Dasher Speed curve suggests that they were approaching this level. Towards the end of the experiment, Dasher no longer increased its speed, because the participants were barely coping with the task with the Dasher Speed setting they had attained.

Error Rates

Error rates were measured in two different ways: in terms of MSD error rate and rate of backspacing (the over-production rate, often measured in KSPC, is not applicable with Dasher, because it is operated with continuous navigation instead of discrete keystrokes).

The error rates fell during the experiment (see Figure 9.8). The grand mean of MSD values for the first session was 10.72%, and the grand mean for the tenth session was 0.57%. The grand mean MSD for the mouse session was 0.93%. The grand mean for average percentage of wrong words in the first session was 33.08%; the equivalent figure for the tenth session was 4.04%.

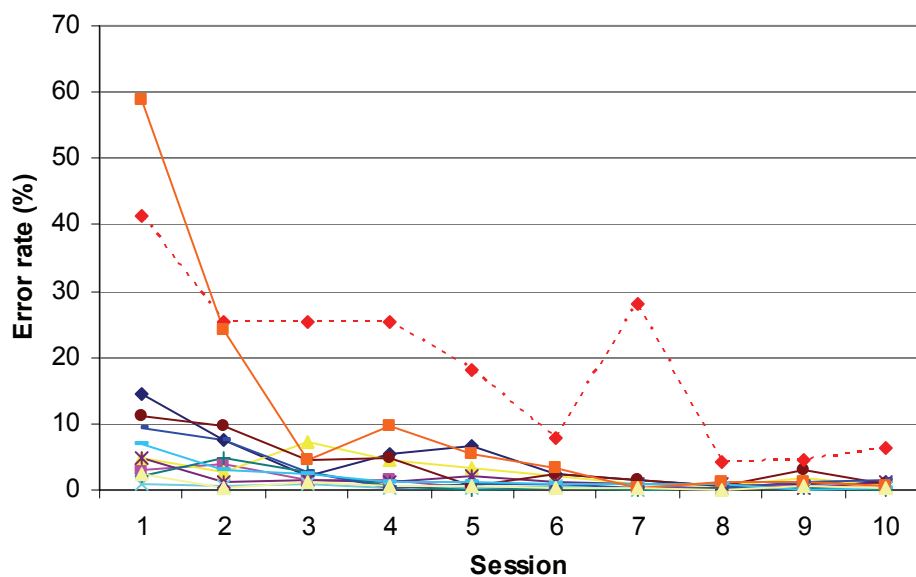


Figure 9.8. MSD error rate (%) for each participant per session.

The rate of backspacing indicates how often the participants canceled characters. Thus, this measure correlates with errors to a degree. The rate of backspacing can be calculated by dividing the total number of characters erased prior to the current position by the total number of characters typed (Itoh et al., 2006). Our participants' rate of backspacing was reduced considerably during the experiment. The grand mean is 0.26 for the first and 0.13 for the tenth session (see Figure 9.9).

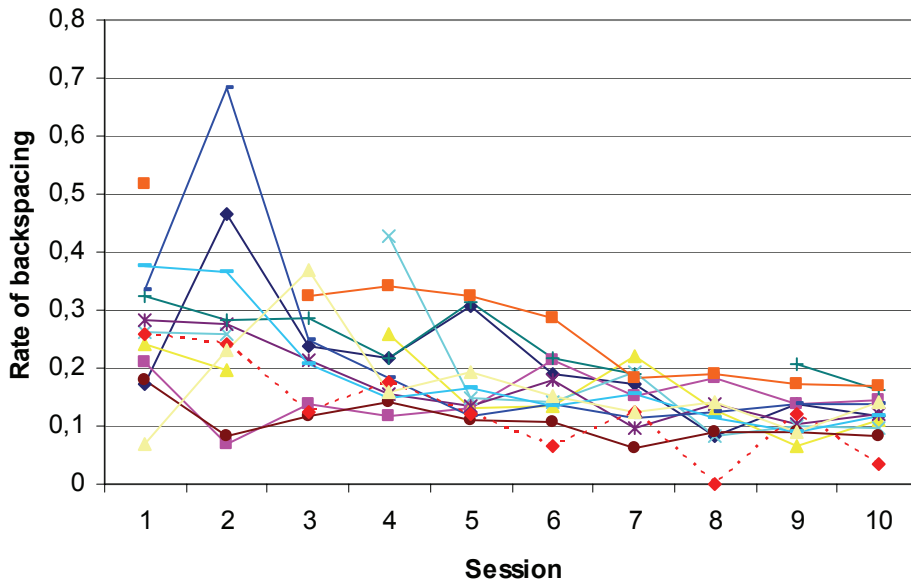


Figure 9.9. Rate of backspacing for each participant per session.

Gaze v. Mouse

Even though the participants had only one session writing via Dasher with the mouse, they were significantly faster with the mouse than with the eye tracker. The participants started the mouse session with the Dasher Speed they had achieved after ending the tenth session by gaze, with the “Adapt speed automatically” option still on. The participants were faster with the mouse, with an average of 20.69 wpm as compared to an average of 17.26 wpm with gaze in the tenth session (see Figure 9.10). A paired-samples t-test showed that this difference was statistically significant ($t(10) = 3.3, p < .01$). The participants also had a significantly higher Dasher Speed with the mouse (3.91 bps v. 2.64 bps) ($t(10) = 3.01, p < .05$). One participant was faster with the eye tracker, and one had the same speed with both devices.

The participants made slightly more errors with the mouse (with an average MSD of 0.94%) than with gaze (with an average of 0.57% in the tenth session). The participants exhibited a lower rate of backspacing with the mouse (0.09) than with the eye tracker (0.13). The differences are not statistically significant.

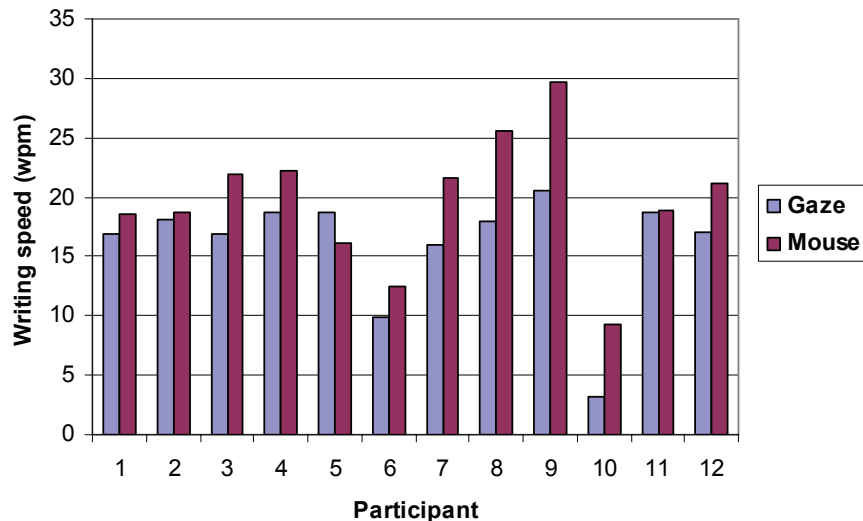


Figure 9.10. Writing speed (in wpm) using Dasher with gaze v. mouse for each participant.

Subjective Experience

Usability is not all about efficiency (speed) and accuracy (error rates). Learnability and subjective satisfaction, for example, are equally important. After the tenth session, we interviewed the participants. They were asked about their preferences, problems that occurred during the experiment, and ideas for improvements. The participants were also free to express their opinions at any time between sessions.

Overall, the participants' comments about Dasher were positive. They felt that they had learned to use Dasher fairly well. Some commented that they had first felt Dasher to be quite hard to use but after a few sessions found the writing to be much easier – and even fun.

All participants were surprised and impressed by how fast the writing with Dasher by gaze actually is. They all – apart from the one outlier – thought that gaze writing with Dasher is much faster than eye typing with an on-screen keyboard, because one does not have to focus one's gaze in one place for so long. Writing with Dasher was, in their opinion, fast also because of Dasher's embedded prediction. They gave comments such as these: "This is really fast," "At the end, writing was faster than I could have imagined in the beginning," and "Because of the prediction, writing is really fast." However, they did sometimes wonder why a word that they thought was not such a common word in Finnish was offered by Dasher more easily than the word they wanted to write.

Dasher's biggest problems from the participants' point of view were the location of punctuation (11 out of 12 participants indicated this to be a problem) and the center circle (noted by four participants). They found it hard to remember where the punctuation characters are situated (there is

no obvious order, such as “abc...,” that would aid in remembering the location or order). Participants had problems both in finding the punctuation marks at all and, especially, in finding the more uncommon marks. They also reported that one can learn quite quickly where the period or the comma is situated but trying to find a more uncommon punctuation mark was quite difficult. The prediction system gave the uncommon marks very low probabilities, which also meant they got very little space when compared to the more common characters, with big rectangles. Participants reported that the center circle was hard to use because it had a relatively long dwell time. One participant suggested that the center circle should have a shorter dwell time. One participant even suggested that the center circle should be bigger so that it would be easier to focus the gaze on it. Obviously, this is easy to correct, and the Dasher developers have informed us that the center circle already is user-adjustable in the Linux version.

All of the participants were novices in writing with gaze, although a couple of them had previously tried an eye-controlled on-screen keyboard briefly. It seemed that some participants had some problems in focusing the gaze (most probably due to inaccuracy in calibration). One participant reported that he had to always look above the correct letter’s position. In the beginning, some of the participants commented that using Dasher was tiring for the eyes, but the effect diminished when they got used to it. Participants also reported that it was harder to write with Dasher via gaze when Dasher Speed grew faster. One participant commented that the faster Dasher moves the more one has to concentrate on writing. Most participants felt that when the speed was faster, Dasher became more unstable (as one participant described it, Dasher “tossed about”) and thus writing was harder in general – and also harder on the eyes.

At the end of the whole experiment, the participants were asked to state whether they preferred the mouse or the eye tracker in Dasher use. Six of them chose the eye tracker, and six chose the mouse. Generally the participants reported that the speed was easier to control with the mouse. One participant who chose gaze said that it felt stupid to write with Dasher by means of the mouse for this reason: If one can use one’s hands, why not write with a keyboard?

Discussion

None of our 12 participants learned to write by gaze with the frequently mentioned top speed of 25 wpm reported by Ward and MacKay (2002). However, one of the participants came quite close, with a result of 23 wpm (achieved in Session 9 – see Figure 9.4).

One reason might be the different language. In Finnish the word endings are inflected a lot more than in English. In Finnish, not only the tense

(present, past, etc.) has an effect, but the inflection is also used for situations handled by prepositional constructs such as “to,” “in,” “for,” and “from” in English. This affects the word prediction process in that the number of potential continuations within a word increases. For example, let us take the word “Finland,” which is “Suomi” in its basic, uninflected form. The Finnish analogues for “to Finland,” “in Finland,” “for Finland,” and “from Finland” are “Suomeen,” “Suomessa,” “Suomelle,” and “Suomesta,” respectively. Practically all nouns are affected by the inflection patterns, including names of persons and places.

Dasher’s prediction capability is only as good (i.e., only as representative or accurate) as the text corpora used to build the model. In our study, we noted that Dasher’s current Finnish language model (based on the modern-language novel *Pereat mundus*, by Leena Krohn (1998)) was not the best possible, because it had some words that are not really Finnish but are still accorded rather high probability. For example, when the participant tried to write something that begins with “H,” such as “Hän” (“She” or “He” in English), Dasher always offered the word “Håkan,” which is a Swedish boy’s name. Because of that, the participants had to correct many phrases that began with “H.” Also, the word “Jumala” (“God” in English) was one that Dasher offered when the participant began to write a word that began with “j” or “J,” even though “Jumala” certainly is not the most common Finnish word to start with “J.” If the training text had better matched the test text, the results could have been slightly better. Furthermore, had the experiment continued, the language model would also have improved automatically since the option that enables the model to adapt (learn) as the user writes was on.

Even though the participants wrote for only 15 minutes with the mouse, they were much faster with it than with the eye tracker. Of course, by that time they were already familiar with Dasher, and obviously there was a strong transfer of the learning effect from the eye-controlled Dasher to the mouse-controlled Dasher. There are several reasons for the mouse performing so much better than the eye tracker, some of which were noted by the participants also. Most of the reasons originate from the features of the human visual system. First, writing with the mouse is easier because the user’s eyes are free to look around and search for the next characters or quickly check the text that has been written thus far. The mouse also is more accurate and does not have any calibration problems. This makes accurate pointing easier. In addition, it increases Dasher Speed because the mouse’s pixel-level accuracy makes it possible to keep the cursor nearer to the right edge of the screen. Dasher Speed increases when the cursor is moved to the right and decreases when the cursor is moved left toward the center, where all movement stops.

One of the participants was much slower than the others – so much so that we discarded his data from the analysis. The problems originated partly from the inaccuracy in the calibration. The participant mentioned that his eyes were getting tired because the cross mark (the measured point of gaze) and the line (Dasher’s zooming direction) did not match. The inaccuracy does not, however, explain all of the problems he had. It seemed he had a hard time grasping (or getting used to) Dasher’s working principles. For example, he complained that “it was hard to perceive [the target letters] because there were so many letters shown around [the desired letter].” He also kept constantly forgetting that the next character should always be selected inside the current character’s box, which prevented the language prediction from working correctly. This may explain why he was the only one who wished Dasher would not use prediction and that the boxes around the letters would remain equally sized. Furthermore, he only rarely exercised the ability to easily cancel by looking left (his backspace rating is low when compared to those of other participants; see Figure 9.9, with the outlier’s figures marked with a dashed red line). His text entry speed was much higher with the mouse than with eye tracking (the mouse does not have calibration accuracy problems), but the error rate increased with the writing speed (he still did not use the cancellation option to correct the errors).

It is important to acknowledge that there indeed can be such “outliers,” people who will need a long time to learn or who may never benefit from Dasher as much as others do. It is interesting that even within the relatively small group of 12 participants there was such an outlier, with an average speed of 3.24 wpm (in the tenth session). In addition, there was a participant whose average speed was 9.82 wpm (all other participants topped out at or above 15 wpm). With a larger, more heterogeneous group, with varying abilities and disabilities, such outliers might be more common.

Conclusion

Dasher is a revolutionary concept for text entry. It has been claimed to display text entry speeds that are almost twice as high as those obtainable via other gaze-based text entry methods. However, the controlled experiments have thus far been small in scale, and the highest text entry rates have been obtained by the developers of the technique themselves.

We wanted to find out how long it takes to learn gaze writing at a high speed with Dasher, and also what the top entry rate would be for a fairly large group of participants (12 subjects). We carried out a longitudinal study that involved 10 sessions of 15 minutes of gaze writing over a period of one month. The subjects did not have prior experience with gaze writing. This resembles the situation of someone suddenly losing the

movement of the other muscles apart from the eyes, and having to learn a completely new means of communication.

The learning curve that we observed was quite exceptional: with the 10 sessions, the increase in text entry rate was still almost linear. After 2.5 hours of practice, participants were able to enter text at an average rate of 17.26 wpm, with the top performer reaching 23.11 wpm. The numbers are lower than we expected, but this is at least partly explained by the quality of the corpus used to build the language model that Dasher used in our experiment. In real long-term use, the adaptation of the language model could yield further speed-ups, in addition to those obtained through the learning of the user. In our experiment, the sentences were biased towards being varying, without much repetition of words, which is not likely to be the case in real life.

An interesting topic for future work would be to obtain a prediction of the top speed obtainable in eye-controlled Dasher. This would require a considerably longer experiment with expert subjects who regularly produce large amounts of text. As judged from the linear growth of the entry rate curve, rates of 25 wpm or higher do not seem unrealistic.

9.3 LEARNING TO TYPE BY GAZE WITH AN ADJUSTABLE DWELL TIME

Introduction

After conducting the longitudinal study on Dasher, we (Majaranta et al., 2009b) became interested in how quickly novices can learn to type by gaze when using the most typical setup: an on-screen keyboard and dwell time. We conducted a longitudinal study to find out how quickly novices learn to type by gaze when an adjustable dwell time is used. We used the same eye tracker (Tobii 1750) and followed the method and procedure used in the Dasher study.

Previous Research

According to Majaranta and R  ih   (2007), and as discussed above, most eye typing evaluations have been conducted with novices using a constant, fairly long dwell time (typically between 450 and 1000 ms). A long dwell time is good for preventing false selections, but a long fixation on the same target can be tiring to the eyes. The dwell time also sets a limit to the maximum typing speed because the user has to wait for the dwell time to elapse before each selection. Reported typing speeds have typically been fairly slow, from 5 to 10 wpm. The entry speed of real experts has not been measured for any of the eye-controlled text entry systems, except in the longitudinal Dasher study reported upon above.

In a more recent study, Wobbrock et al. (2008) compared dwell-time-based gaze typing with gaze-gesture-based text entry in a longitudinal study (14 sessions with eight trials in each). They used a short dwell time, 330 ms; nevertheless, their result of 7 wpm is in line with previous research. For experimental reasons, Wobbrock et al. restricted the size of the on-screen keyboard to match the fairly small window of the gaze-gesture-based system, which may explain the comparatively slow typing speed; small buttons are hard to hit by gaze.

Špakov and Miniotas (2004) studied automatic adjustment of dwell time. Even though their results were encouraging, there was some delay and involuntary variation in the automatic adjustment. The participants would have wanted more control over the dwell time adjustment – for example, to be able to change the speed more quickly. Therefore, they suggest a tradeoff on the extent of the automatic control, to let the user decide when the dwell time is convenient.

We conducted a longitudinal study to find out how quickly novices learn to type by gaze when allowed to adjust the dwell time at will as they see fit. The method and a summary of the results are reported below.

Method

Participants

Eleven able-bodied university students volunteered for the experiment (three males and eight females, from 18 to 30 years of age, with normal or corrected-to-normal vision). All were native speakers of Finnish and familiar with the QWERTY keyboard layout but were novices in gaze typing. All participants were rewarded with four movie tickets. To motivate the participants in the 10-day experiment, we informed them after the first session that the participant learning the best to gaze type (in comparison of final performance to each participant's own initial results) would receive an extra prize.

Apparatus

The Tobii 1750 gaze tracking device, integrated with a 17-inch TFT color monitor (with 1280 x 1024 resolution), was used to track the participants' gaze.

The COGAIN ETU Driver, with a plug-in for Tobii, was used to implement the experimental keyboard and to save data. The stimulus phrase was shown on top of the experimental keyboard (illustrated in Figure 9.11). The transcribed text written by the participant appeared in the text input field below the stimulus. Letters were organized into a QWERTY-like layout, including keys for the most common punctuation.

Space, Shift (for uppercase letters), and Backspace were located below the letter keys. The last row included the keys for adjusting the dwell time (in the middle) and a “Ready” key (on the right).

We decided to use a speed meter as an indicator of the typing (or selection) speed instead of (a numeric) dwell time adjustment, because the former was considered more natural and easier to understand for the users. The gaze-operated minus key decreased the speed by increasing the dwell time (max.: 2000 ms), and the plus key increased the speed by decreasing the dwell time (min.: 150 ms). When the speed indicator’s pointer was in the middle, the dwell time was 600 ms. Thus, the steps to adjust the speed became smaller as the indicator moved to the right. This enabled a rapid increase of speed with long dwell times and fine adjustment of speed with very short dwell times. The formula for the dwell time adjustment was based on pilot tests: $DT_{adjusted} = 300^{(X/12)} - 150$, where $X = \{0,1,\dots,24\}$. X is the step controlled by the user using the minus and plus keys, which change the X value by 1. At the lower end, when the dwell time duration is long, the step is 160 ms, and at the higher end (with very short dwell time) the step is only 25 ms.²²

An animated closing circle was shown on the key to indicate the progression of dwell time (see “i” in Figure 9.11). The color of the animation was chosen such that it offered as little disturbance as possible but was still easy to see.

When the dwell time ran out (the circle closed), the key was visually depressed and a “click” sound was heard. The participants were told that they could ask the experimenter to remove the animated feedback if they found it disturbing.

The active selection area was bigger than the visible key (covering the full area between the visible keys) in order to minimize potential problems caused by inaccuracy in calibration. Thus, the key was selected (and feedback shown on the desired key) even if the measured point of gaze was somewhat outside the key.

²² Values produced by the formula were rounded for clarity (e.g., 176 ms was rounded to 180 ms).

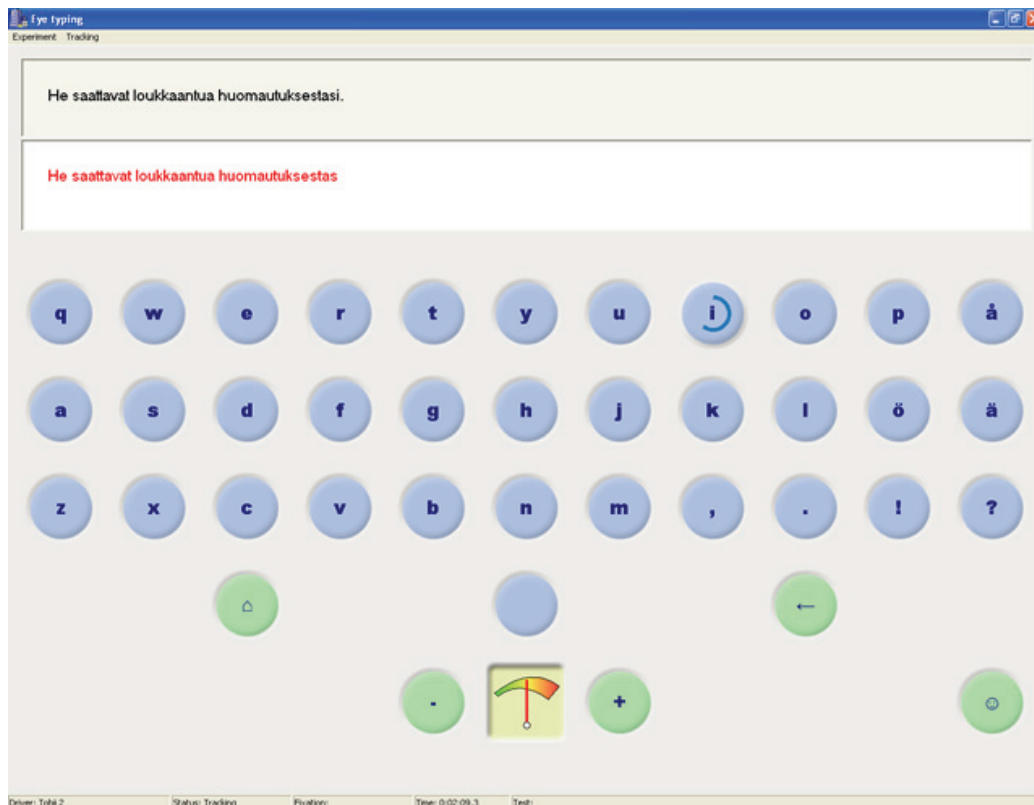


Figure 9.11. Experimental software. A shrinking circle is shown on the letter “i” as an indication of the progression of dwell time.

Procedure and Design

Each participant was first briefed about gaze control and the motivation for the study. Before the actual test, the participants practiced gaze control briefly by playing three rounds of a simple board game (Tic Tac Toe) by gaze.

The participants were seated such that their eyes were approximately 50–60 cm from the monitor. They were instructed to sit fairly still, but their movements were not actually restricted in any way. The gaze tracker was calibrated at the beginning of every session. Re-calibration was carried out if needed, but we tried to do it between phrases. If the tracker had to be re-calibrated in the middle of a phrase, that phrase was ignored in the analysis.

The task was to type as many phrases as possible within the 15-minute time limit. The phrases were the Finnish translation (Isokoski & Linden, 2004) of the 500-phrase set originally published by MacKenzie and Soukoreff (2003).

The phrases were shown one at a time. After finishing the phrase, the participant selected the Ready key, which loaded the next phrase. The software was set to stop after the 15 minutes had elapsed and the

participant had finished typing the last sentence. The timer ran only during active typing, starting from the entering of the first letter and ending with the selection of the Ready key.

Participants were instructed to memorize the phrase first and then to write it as quickly and accurately as possible. They were told to correct errors only if they detected them soon after the error occurred (that is, within the last word).

The dwell time was initially set to 1000 ms for all participants. Participants were instructed to adjust the dwell time between sentences, but they were able to adjust it at any time they wanted to do so. The use of special keys and the rules for correcting mistakes were explained in every session.

Each participant visited the laboratory 10 times. The sessions were organized such that there was never more than two days between consecutive sessions, though there were a few exceptions involving three days between sessions.

The first and the last session took about an hour each, with initial preparations, instructions, and final interviews. The other sessions lasted about half an hour, including preparations and a short questionnaire before and after each test. In total, each participant gaze typed for two and a half hours (10 x 15 min).

Results

The results are based on data from 10 participants. One participant was a clear outlier, excluded from the statistics but included in the figures (marked with a red dashed line). Technical problems caused us to lose the data for one participant for one session. The missing values were replaced with an average of those for the previous and the next session. Analysis of a phrase started with entering of the first character and ended with the selection of the Ready key.

Typing Speed and Dwell Time

The grand mean for the text entry rate was 6.90 wpm in the first session and 19.89 wpm for the last, tenth session. Thus, significant learning ($F_{9,81} = 93.60, p < .0001$) had occurred (see Figure 9.12).

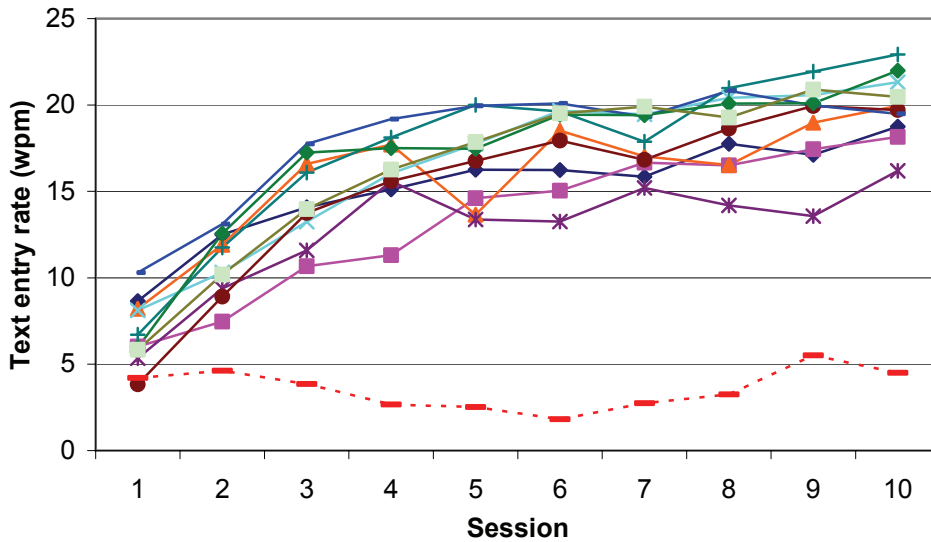


Figure 9.12. Text entry rate in words per minute for each participant per session (the outlier is marked with a red dashed line).

The dwell time was initially set to 1000 ms. The grand mean for the dwell time was 876 ms in the first session and 282 ms for the last. The decrease in dwell time was especially rapid during the first three sessions (see Figure 9.13).

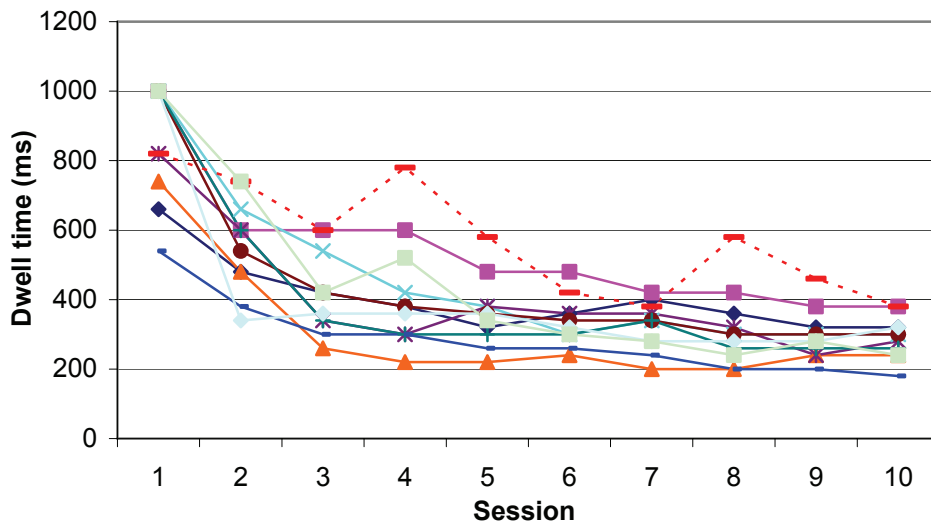


Figure 9.13. Dwell time duration in milliseconds for each participant per session.

Nobody used the minus key (to slow down typing by increasing the dwell time) in the first session, but its use increased in later sessions, with participants making minor adjustments in both directions to find the highest manageable typing speed. In total, the plus key was selected 229 times and the minus key 96 times. During any one session, most participants typically only made a few adjustments (see Figure 9.14); the

grand average for the number of presses of the plus key is 2 (median: 2) and for the minus key 1 (median: 0), ranging from no presses to a maximum of six presses²³ during a session.

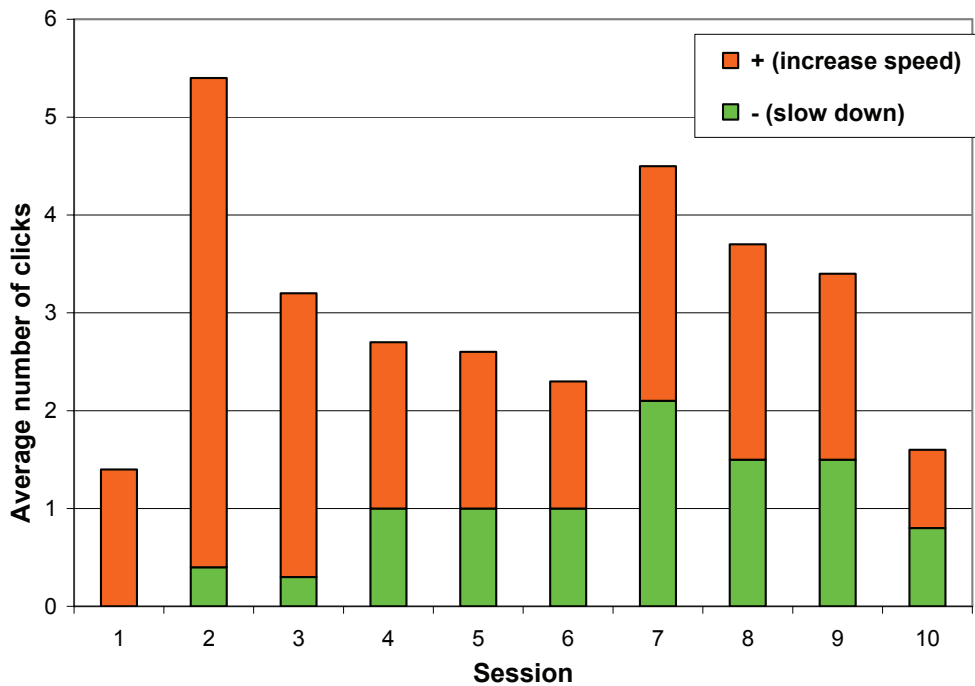


Figure 9.14. Average number of presses (across all participants) of the plus and minus keys on the speed dial.

Accuracy

The grand mean for the MSD error rate was 1.28% in the first session and 0.36% for the last session (see Figure 9.15). Thus, the error rates decreased even though the typing speed increased. Overall, the error rate remained quite reasonable (below 5%) throughout the experiment.

²³ Two outlying measurements have been removed from calculation of the maximum number of presses: in one case, a participant erroneously pressed the minus key seven times, and in another case another participant pressed the plus key 11 times. However, in most cases, people only made minor adjustments, with one or two presses of the plus or minus key per session.

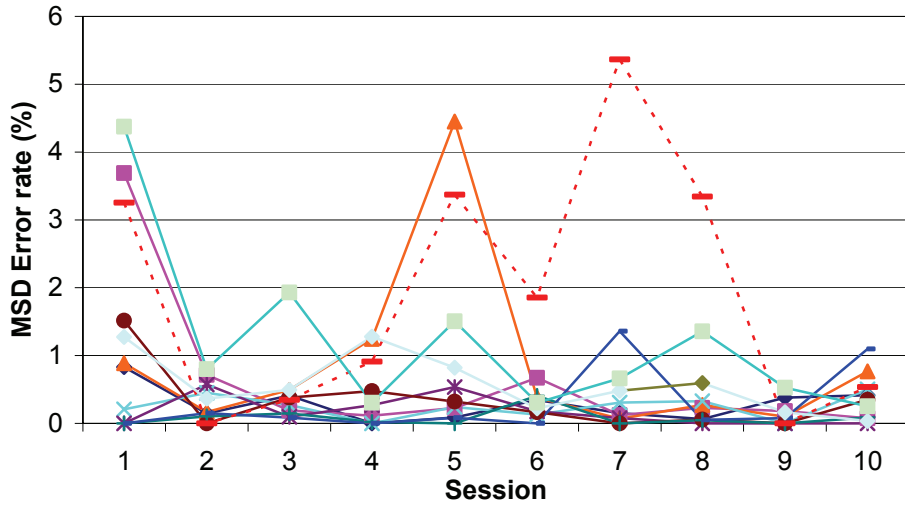


Figure 9.15. Error rate (%) for each participant per session.

The grand mean value for KSPC was 1.09 in the first session and 1.18 in the last session (excluding keystrokes for the Shift key). The slight increase (see Figure 9.16) is not surprising, since there always is a tradeoff between speed and accuracy in text entry tasks. The increase in the KSPC value suggests that people had to correct more errors when the typing speed increased; however, the increase was not statistically significant ($F_{9,81} = 1.13, p > .2$).

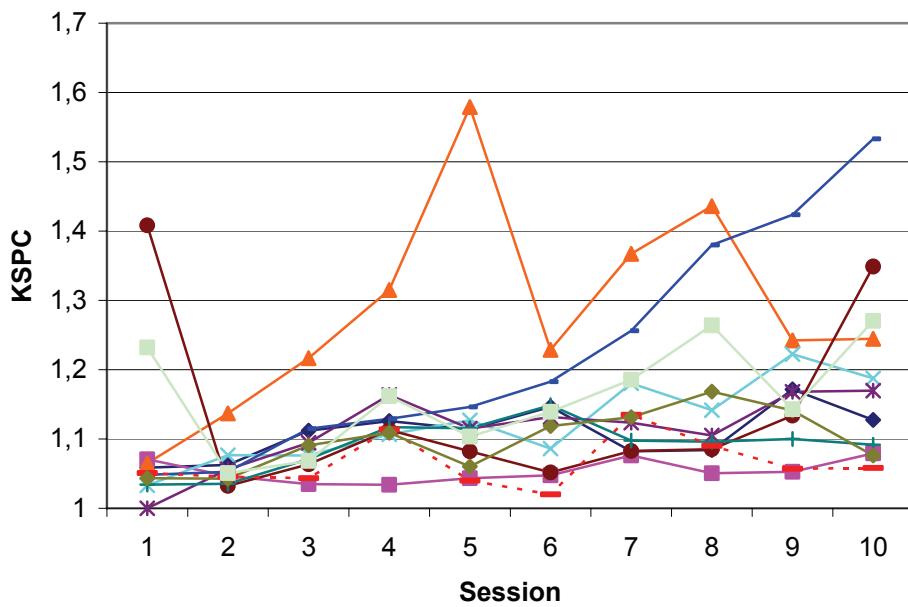


Figure 9.16. Keystrokes per character for each participant per session.

Subjective Impressions

We analyzed the subjective ratings with the nonparametric Wilcoxon matched pairs signed ranks test. In order to measure eye fatigue, we asked the participants how tired their eyes were before each test, and again after the test, on a scale of 1 to 7. The fatigue level was calculated by subtracting the first (“before” test) value from the latter value. In most cases, the participants felt their eyes were (slightly) more tired after the test, compared to before the test session. However, the level of tiredness did not increase in time (i.e., from earlier to later sessions) when the typing speed increased. The average level of the difference in tiredness ratings was 0.6 for the first session and 0.8 for the last session, but the increase is not significant.

We also measured perceived speed and ease of use after each session, using a questionnaire with a scale of 1 to 7 (from “very slow” to “very fast” etc.). There was an increase in the perceived speed (from 4.2 to 5.5, $p < .005$), which is in line with the increase in the measured speed. The perceived ease of use (with an average rating of 5.3) and general fatigue (with an average of 3.5) remained approximately at the same level, showing no significant change over time.

Finally, we interviewed the participants after the last session. Participants felt that typing by gaze was fairly easy, easier than they had imagined, but clearly slower than using a conventional, hand-operated keyboard. Participants appreciated the QWERTY layout because of its familiarity. All participants felt they had improved in gaze typing from session to session, especially in the beginning.

When we asked whether there was something that was especially exhausting to the eyes, a couple of participants commented that the high level of concentration required and not being able to blink normally were exhausting to the eyes. A couple of other participants felt the brightness of the screen was tiring to the eyes.

All participants felt the typing speed adjustment clear and easy to use. They felt that they had enough feedback on the gaze-controlled selection of a key. Auditory feedback was considered either more important than (by six participants) or equally important to (by three participants) the visual feedback. Participants also appreciated the animated feedback and wanted to keep the closing circle even with very short dwell times; five participants tried gaze typing without the circle, but only two had it turned off at the end of the last session.

Half of the participants experienced problems in using the Shift key: with short dwell times, participants experienced a delay in screen refreshing when lowercase letters were changed to their uppercase versions. This caused disorientation and difficulty in selection of the next letter. Some

participants accidentally selected the Shift key instead of the Space key or vice versa, and one participant suggested that the latter should be shown as a wide bar similar to the spacebar of a conventional keyboard. Some participants also experienced occasional problems with keys that were located near the edges of the screen, because of the decreased accuracy of the eye tracker calibration in those areas.

Discussion

All participants (including the outlier) adjusted their dwell time such that it was below 400 ms in the last session (min.: 180 ms, max.: 380 ms). The decrease was especially rapid during the first few sessions; thus, in the fourth session, the average dwell time was already down to 378 ms. Correspondingly, the average typing speed had increased from 6.9 wpm to 16.2 wpm by the fourth session, with a reasonably low error rate of 0.37%. Four 15-minute sessions equal one hour of practice, after which time the learning decelerated considerably. Therefore, one should not draw any conclusions on the typing speed or concerning the efficiency of the dwell-time-based gaze typing system before the user been able to practice for long enough (at least one hour), and, as importantly, has been able to adjust the dwell time accordingly.

This study with QWERTY followed the method used in the Dasher study (described above in Section 9.2), with an equal amount of practice and similar test procedures. Dasher (Ward & MacKay, 2002) is considered the world's fastest method for writing by gaze. Therefore, it is noteworthy that the final typing speed of 19.9 wpm in this study is comparable with the results in the Dasher study with its average of 17.3 wpm in the tenth session. However, it should be noted that in the Dasher experiment, the speed curve was still growing rapidly after the 10 sessions, suggesting a potentially significant increase in speed even after the 2.5 hours of practice (see Figure 9.17). In addition, the average error rates for Dasher were slower than for QWERTY. Learning to master Dasher obviously takes more time, but with further practice, users of Dasher are likely to exceed the text entry speeds obtainable with an on-screen keyboard.

It is hard – and probably unfair – to compare these totally different text entry methods. However, our results do show that people can type by gaze fairly rapidly and accurately by using a simple, easy-to-learn on-screen keyboard, provided that fixed dwell time does not slow down the typing (and provided that careful consideration is given to the precise control-feedback loop).

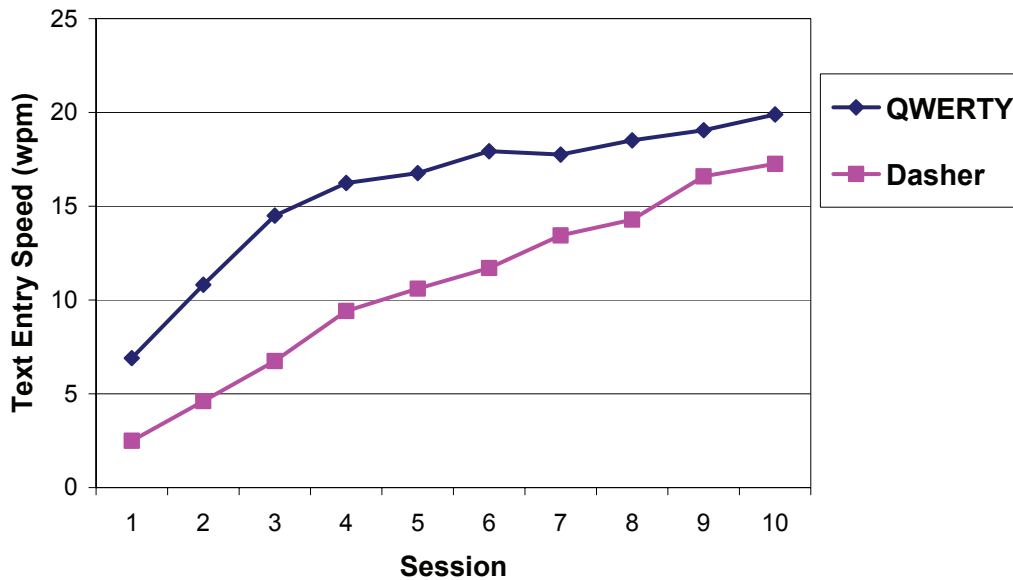


Figure 9.17. Dasher v. QWERTY.

So, in theory, how quickly could one type when using an on-screen, dwell-time-operated keyboard? Figure 9.18 illustrates the theoretical maximum typing speed with a 50 ms²⁴ search time, compared with the average typing speed with QWERTY as measured in our experiment. Obviously, 50 ms is very brief, and the actual search time (required for a saccade from one key to the next) varies, depending on cognitive processing and the distance between the targets (though this effect is very small when compared to other factors). Part of the difference is explained by the need to correct more errors, since the increase in speed typically causes an increase in the number of errors (note the slight increase in KSPC). We also know that the feedback may have an effect on performance, for people may take more or less time before proceeding to the next key (see Chapter 8). For all these reasons, it would be interesting to repeat the study by Salvucci (1999), described in Section 6.1, as it is, without any feedback; it would be interesting to experiment with the fastest pointing speed obtainable without any potentially confounding factors.

²⁴ This extremely short search time of 50 ms is used here only for theoretical calculation purposes.

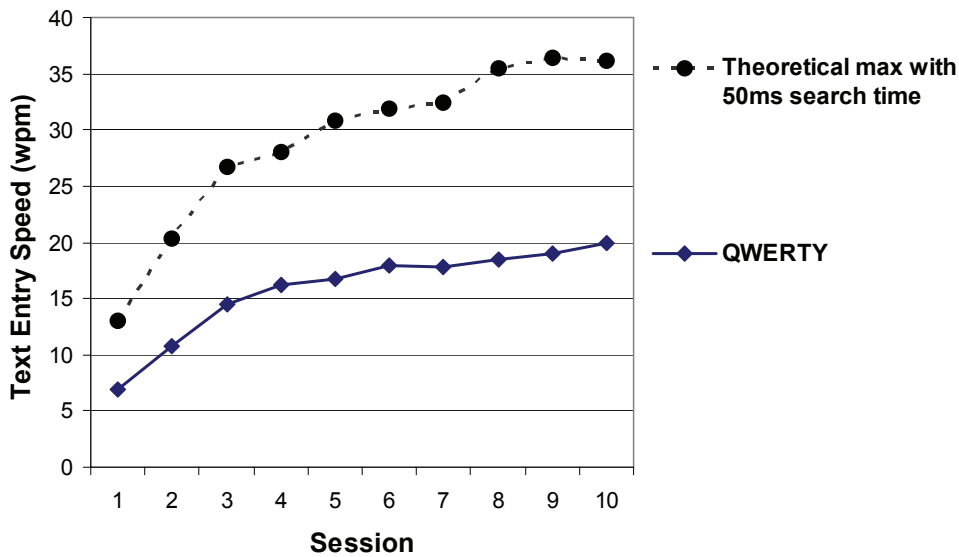


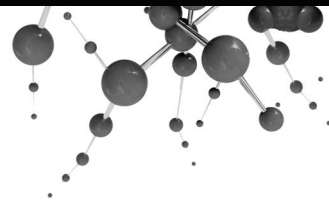
Figure 9.18. Measured typing speed v. theoretical maximum with 50 ms search time.

It may be interesting to experiment to determine the maximum typing speed. However, in practice, the typing speed that is convenient and comfortable to the user varies, depending on the cognitive and emotional state of the user. In our study, the participants tried to type as rapidly as possible (following our instructions and trying to win the extra prize). Nevertheless, a few of them commented after some sessions that the high speed required a high concentration level and made them feel exhausted after the session. A couple of participants also commented on their current condition and assumed it might affect their performance. For example, one participant felt tired because of not having had enough sleep the night before the test. Another participant commented that gaze typing requires high concentration and that a break in concentration causes errors. Hence, we recommend that adjustment of dwell time be made easily available for the user. This may seem an obvious feature to include. However, according to our experience with currently available gaze communication systems, the dwell time adjustment is typically “hidden” in the setup or options dialog. In some systems, the options dialog has not been made available for gaze control but can only be operated with a hand mouse (or via mouse emulation).

In the final analysis, how quickly can the real experts, actual users of eye control, enter text by gaze alone? The *MDA/ALS Newsmagazine* (Quintero, 2009) ran a story about Jack Orchard, who has written a whole book, his autobiography, with the Eyegaze system from LC Technologies. According to the magazine, he is able to enter text at a rate of about 30 to 35 wpm, using a dwell time set to 180 ms. He uses a speed on-screen keyboard in which letters are arranged around the space bar according to their frequency in the English language. This layout enables fast text entry with minimal eye movements (Quintero, 2009).

Conclusion

We conducted a longitudinal study to find out how rapidly novices learn to type by gaze when an adjustable dwell time is used. Our results show that the text entry rate increased from 6.9 wpm in the first session to 19.9 wpm in the tenth session. Correspondingly, the dwell time decreased from an average of 876 ms to 282 ms, and the error rate decreased from 1.28% to 0.36%. The typing speed achieved, nearly 20 wpm, is comparable with the result of 17.3 wpm obtained in an earlier, similar study with Dasher. Thus, we conclude that people can gaze type fairly quickly and accurately when using a simple, easy-to-learn on-screen keyboard, provided that fixed dwell time does not slow down the typing.



10 Moving from Text Entry to Editing by Gaze

10.1 EDITING TEXT BY GAZE

Gaze-based text entry systems typically provide a backspace or undo key for immediate corrections. However, since the keyboard itself takes a lot of space, there is not much space left for editing commands (such as copy, paste, bold, and underline). The editing commands are therefore often hidden in the virtual keyboard's menu structure.

Furthermore, if the user wants to place the cursor (caret) in a certain location in the text to correct a spelling mistake, the caret may land a few characters (horizontally) or a couple of lines (vertically) away from the desired location as a result of calibration accuracy problems or drifting. For this reason, many systems provide navigation buttons (left, right, up, and down) for adjusting the cursor position.

Using the same modality for both input and output presents another kind of challenge for the interface design. Since the gaze is needed for selecting the button, the user cannot see the effect of editing on the text simultaneously but needs to leave the keyboard area to review the result of the action on the text.

We (Majaranta et al., 2009c) developed a dynamic pie-like²⁵ menu that can potentially facilitate the task of text editing by gaze. A pie menu is a pop-up menu that appears at the place of focus. The menu items are placed in a circular pattern around the center of the pie. Pie menus have been proven useful in mouse-based interaction (Kurtenbach & Buxton, 1994). Our assumption is that having the editing commands near the focus of interest can facilitate the process of editing by gaze. In this section, we first briefly review related research. We then introduce our prototype of a dynamic pie menu for text editing by gaze and report preliminary results from our first pilot study, in which we compared the dynamic pie menu with a static editing menu. Finally, we discuss ideas for further improvements and future work.

Related Research

Pie menus have proven to be faster than ordinary linear menus in normal mouse-based interaction (Kurtenbach & Buxton, 1994), and they have also been successfully utilized for gaze-based text entry (Huckauf & Urbina, 2007; introduced in Section 5.3).

Other related research includes the work by Tien and Atkins (2008) who tested different menu layouts for gaze interaction: a layout that resembles a typical drop-down menu, a layout resembling typical gesture-based menus found in hand-held devices, and a variation of the gesture-based menu adjusted for gaze – with big buttons and short distance. They did not find significant differences in task times between the layouts. After the initial experiment, they implemented several improvements for the menu designed specifically for gaze, such as a “snap-on” feature that snapped the eye mouse cursor to the center of the button and a feature that opened the menu with a quick off-screen glance to the left. In the follow-up experiment, they found that, after memorizing the menu commands, participants were able to perform menu selections by using dwell times as short as 150 or 180 ms.

Also Kammerer et al. (2008) experimented with three different designs of multi-level menus operated by gaze. They found that a semi-circular menu was better suited for selection by gaze than a full-circle layout or a linear (conventional) menu design. The semi-circle had the sectors located only on one side of the menu, which probably made it clearer than the full circle menu and therefore easier to perceive and navigate (with sub-menus

²⁵ We admit that our implementation of the pie prototype does look like a pie menu. However, the ultimate goal is to develop an “iPie.”

also extending on one side only). In view of participants' subjective experience, Kammerer et al. stated that the major drawback of the full-circle menu was its confusing arrangement (widespread and ungrouped menu items) and the long distances between menu items.

Dynamic Pie Menu for Text Editing by Gaze

We implemented a prototype of a gaze-operated dynamic pie menu for text editing (illustrated in Figure 10.1 on the top). The pie menu is shown at the point of the user's focus when the user fixates on the text for longer than the predefined dwell time (1500 ms). The cursor (caret) is located in the center of the pie. The user can see the text through the central hole, and also the menu items are partially transparent.

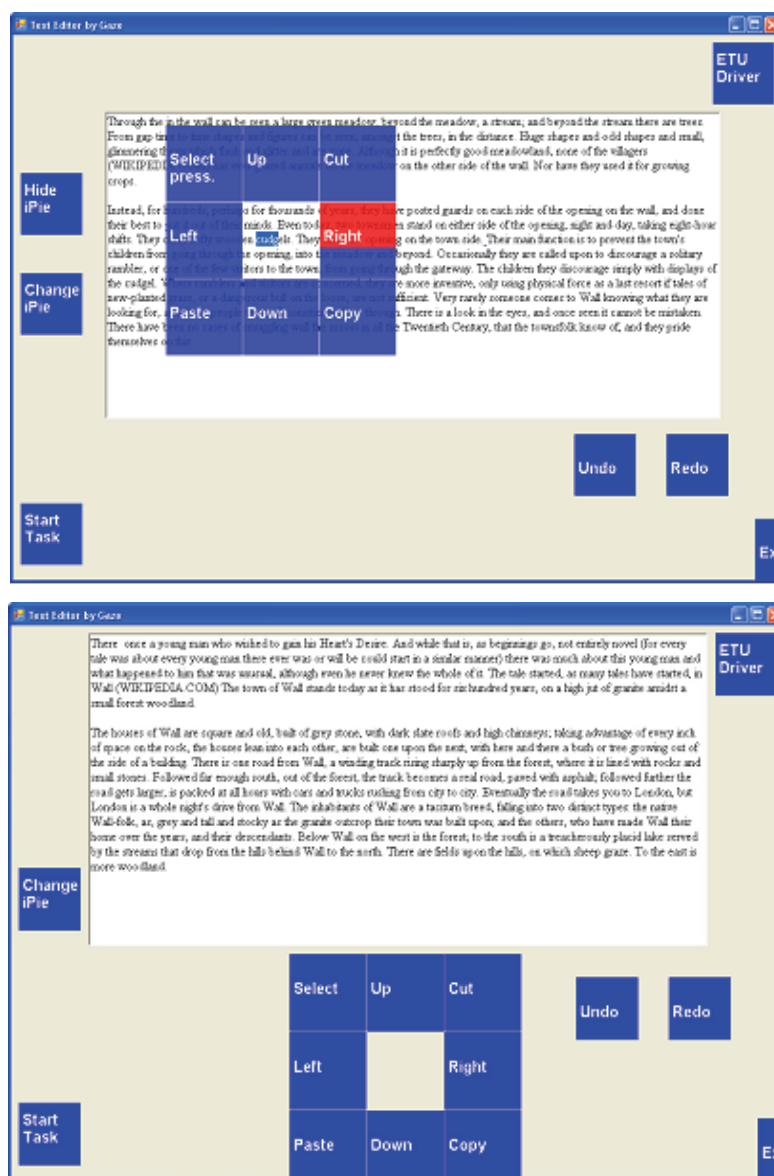


Figure 10.1. Dynamic pie menu (top) and static menu (bottom). The image on the top also illustrates the feedback given on dwell time progression for selection of “Right.”

The user can fine-tune the (often misplaced) cursor position by using the left, right, up, and down keys. The pie menu moves along such that the cursor is always in the center. To select text, the user needs to press (dwell on) the Select button and then move the cursor with the direction keys. An editing command such as “Copy” can then be executed for the selected text. Our experimental prototype included functions for basic text editing (cut, copy, and paste) and text formatting (bold, italic, and underline). The commands in the pie menu can be changed by using the Change iPie button (at the left side of the application window).

In addition to the dynamic pie menu, we implemented a static menu that remains in a fixed location on the bottom of the screen (see Figure 10.1, bottom image). The layout and functionality of the keys were the same for the two menus. The dwell time for selecting a button was set to 1000 ms for both. If the user kept on looking at the button, it started to repeat the click, with an interval of 450 ms. Durations are based on pilot tests.

Method

We conducted an initial feasibility study with 13 participants (10 males, three females, age spread of 19–26 years, mean age of 21) to learn about the potential usefulness and usability of the dynamic pie menu for text editing by gaze. All participants were university students with good computer skills and average to good text editing skills. All were novices in editing text by gaze, but two had some experience of gaze control and one had participated in an eye-tracking-related experiment. The Tobii 1750 eye tracker was used with the COGAIN ETU Driver to track the gaze.

The experiment was a within-subjects study with two conditions: dynamic pie menu and static menu (as illustrated in Figure 10.1). The participants were assigned to two groups; participants in the first group started with the dynamic pie menu, and participants in the second group started with the static menu.

The participants were first briefed on gaze interaction and the experiment. They then filled in a pre-experiment questionnaire. Each test started with calibration. Before the actual test, the experimental software was introduced to the participants, starting with the condition that was assigned to the participant, and they had a chance to practice using it with two simple tasks and to ask questions.

During the test, each participant performed six similar tasks with both interfaces. Each task started with the press of the Start Task button and was ended by selecting “End Task.” Participants started with four simple formatting tasks – for example, selecting a word and boldfacing it. The last two tasks were text editing tasks, wherein the participant had to move a word or to swap two words by using the cut and paste commands.

After completing all six tasks with one condition, the participants were interviewed about the first design. This procedure was repeated with the second condition, starting with the introduction of the interface and practice, and ending with the interview (the same questions were asked of all participants with both interfaces). After finishing the tasks in both conditions, the participants filled in a questionnaire where they had a chance to compare the two designs and we interviewed them.

Preliminary Results

We lost data from several tasks from several participants as a result of technical problems. A few participants had poor calibration, which affected their performance. In addition, there was a bug in the experimental software that we noticed only after the tests had begun. Therefore, we will not report statistically significant results for the performance measurements; instead, we focus on reporting initial user reactions and ideas for further improvement.

Despite losing some of the data, we did look into task completion times (including only successfully finished tasks with no bugs). There seems to be a trend in the task times indicating that the participants performed faster in the simple formatting tasks (tasks 1–4) by using the dynamic pie menu (with an average grand total of 38 seconds) as compared to the static menu condition (with an average grand total of 47 seconds). However, when completing the more complex editing tasks (tasks 5–6), they performed more slowly with the dynamic pie menu (with an average grand total of 77 seconds) than with the static menu (with an average grand total of 67 seconds).

Out of the 13 participants, eight preferred the static menu over the dynamic pie menu (which was preferred by five) if required to choose only one. If they had a chance to use both, five would still prefer using the static design only, three would prefer the dynamic, and five would like to use both (especially after further practice, as noted by a couple of participants).

We were interested in assessing the usefulness of having the navigation keys in the dynamic pie menu, since we assumed they would be especially useful for adjusting the location of the cursor in the text, even if all other functions (formatting and editing) were placed in a static menu. We asked the participants if they felt it was easier to use the arrows in the dynamic or the static menu. Five participants felt it would be best to have the arrows in the static menu (to avoid confusion, as some of them commented). Others felt that placing the arrows in the dynamic menu was indeed a good idea (four participants), or probably a good idea (four participants).

We also asked the participants which of the designs was faster, easier to use, more comfortable, and easier on the eyes. The dynamic and static design received an equal number of votes as faster (6/6, plus one “cannot say” reply). The static design received more votes in all other categories: number of votes for static / dynamic / cannot say: easier 9/3/1, more comfortable 7/5/1, easier on the eyes 7/3/3.

During the interview, we asked what was most difficult in using each of the menus. For the dynamic pie menu, three participants felt that the menu disturbed visibility (of the text under it or the interface in general), and three felt it was difficult to move the gaze between the menu buttons and the text (to see the effect of selection, for example). For the static menu, four participants felt that switching between the menu and the text was difficult, and three felt that placing the cursor in the correct place was difficult. Other difficulties observed by more than one participant were related to gaze interaction in general, such as the difficulty of fixating in the same location for long enough or a feeling of being rushed when the dwell time was running out, or difficulties related to the implementation of the experimental software. For example, we had implemented a feature that automatically hid the dynamic menu if the user looked at the gray area around the text field for longer than the threshold time. However, there was a bug in the implementation: the feature sometimes caused disappearance of the dynamic menu in the middle of text editing.

For both designs, a few participants complained that it was hard to select a full word and to remember how many times they had to eye press “left” or “right” to select all of the letters in the word. Some participants suggested that there should be an option to select a full word, or an option to define the starting and ending point for the selection (instead of repetitive presses of the left or right arrow). Even though most participants felt that the static menu was easier to use and perceive, since it stayed in a familiar location, they also wished it were nearer the text. Several participants wished they could adjust the transparency level of the dynamic menu’s buttons. Other suggestions for improvement of the dynamic menu included replacing the buttons’ caption text with icons, having more options (more buttons or sectors for the pie), and placing the Change iPie button (which toggles between the formatting and editing menus) in the dynamic menu itself for easy access. Some also felt that the buttons were unnecessarily big and too far apart.

Discussion and Future Work

Even though most participants preferred the static menu over the dynamic pie menu, we believe there is potential in having the editing commands in a dynamic pie menu. First, we observed a trend toward faster task completion times when one was using the dynamic pie menu for simple formatting tasks. Second, some participants preferred the dynamic pie

menu, and several more would like to have both options available. It is worth noting that there were more bugs in the dynamic pie menu condition than in the static menu condition, which may have affected the participants' subjective experience (even though we asked them to ignore the bugs in their ratings). The experiment was very short, and participants were novices in gaze interaction; therefore, we believe difficulties related to gaze interaction in general may have affected the results also. More practice would be needed to see the full potential of both solutions. Thus, we plan to organize a longitudinal experiment after improving the design (and correcting the bugs, obviously).

We agree with the participants that the text in the menu button areas should be replaced with icons. Icons would be fast to recognize and would not detract from the visibility of the text as much as the current design, which has partially transparent text buttons over the body text.

In the current implementation, the buttons in the dynamic pie were basically normal dwell-time-activated buttons. With the future design, we want to test a menu that looks like and operates more in the manner of a pie menu: it would be circular with sectors near each other. The sectors could also be selected by simply looking beyond the outer edge of the sector (similarly to the pie menu design by Huckauf & Urbina, 2007): as long as the user is viewing the command icon in the sector, nothing would happen, but as soon as the gaze crosses the sector's outer edge, the sector would be selected - or a new sub-menu (with sub-sectors) would be opened. For example, the basic layout could have the arrows (as icons) and other sectors for formatting and editing. Those could open the next level of commands; for example, activating the formatting sector could show a sub-menu for boldface, italics, and underlining.

Our prototype did not allow using the dynamic pie menu near the edge of the screen, which is why a fairly large empty (gray) area was added around the text field. This problem could be solved by implementing the half-circle layout suggested by Kammerer et al. (2008). Dynamically changing the orientation of the half- (or partial-) circle layout could easily compensate for the lack of space in one direction.

To our knowledge, editing text by gaze has not been studied before. Our research is the first step towards more user-friendly text editing by gaze. We believe this area offers a rich set of opportunities for future research and development.

10.2 FUTURE RESEARCH: INVOLVING USERS WITH DISABILITIES

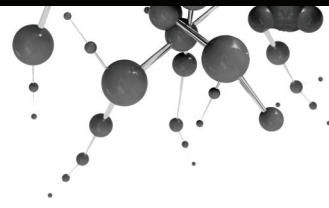
In addition to text entry by gaze, another important direction for future research is to better involve users with disabilities. Apart from a few exceptions,²⁶ most experiments in gaze typing have been conducted with able-bodied participants. Although the results give information on usability and learning processes involved in gaze control, they may not be directly applicable for people with disabilities. For a profoundly disabled person who does not have prior experience of any method of computer control, it may take anything from a few weeks to years to master a gaze control system (Donegan et al., 2006b; Gips et al., 1996). The process can be facilitated by progressing in small steps and carefully considering the current (and evolving) abilities and needs of the user, and by involving the user in the design process (Hornof, 2008). More information on successful eye control assessment and take-up are available in the work of Donegan et al. (2009), and practical hints on how to proceed can be found in the “User Involvement” section of the COGAIN Web portal (<http://www.cogain.org/>).

Organizing controlled experiments with people with disabilities may be problematic on account of their varying (dis)abilities and medical conditions (Aoki et al., 2006). For example, conventional usability evaluation methods may not work as they are; thus, the researchers should be prepared to adjust the evaluation methods to suit the characteristics of the participants (Lepistö & Ovaska, 2004). Furthermore, it may not be practical or even safe to transfer users with severe physical disabilities to a usability laboratory. For rare conditions, the potential participants are few in number and spread far apart. Therefore, automated or remote usability evaluations have been suggested. Remote evaluation would enable collection of large quantities of data in an ecologically valid way in the user’s normal environment and would enable benchmarking of eye tracking or gaze typing systems in actual use.

Aoki et al. (2006) provide a brief introduction to remote evaluation and suggest measures for text entry evaluation that would preserve the user’s privacy by not revealing the content of the user’s communication but

²⁶ Results (e.g., Donegan et al., 2006a, 2006b) from user trials conducted within COGAIN, the European Network of Excellence on Communication by Gaze Interaction, are available on the project’s Web site at <http://www.cogain.org/>. The network combines the efforts of researchers, manufacturers, and user organizations for the benefit of people with disabilities (Bates et al., 2006). I and many of my co-authors are members of the network.

would still transfer useful information about progress. As a general rule, they suggest, the calculation of performance metrics should be done on the local computer and only the result should be sent to the remote machine. A potentially useful measurement that does not reveal the content of the message is the number of deleted characters (or presses of the backspace key). Another potentially useful figure is the number of attended keys not selected. As discussed in Section 9.1, AKNS reflects the search process in the early stages of learning: before the user has learned the layout of the keyboard, he or she needs to review several keys before finding the correct one. An experienced user knows the locations of the letters by heart and can directly point at the correct letter. Aoki et al. compared AKNS with the measures for typing speed (per character) and error rate and found a high correlation of AKNS with error rate. This indicates that AKNS has potential in measurement of the progress of learning.



11 Summary and Conclusions

In this thesis, I reviewed research related to text entry by gaze and presented results from several experiments that studied various aspects of gaze-based text entry. The literature review showed that, even though gaze-based text entry has existed and been in use for decades, the design issues have not been studied in detail. Therefore, text entry by gaze provides a rich set of issues for study both from the practical and from the research point of view.

The eye is a perceptual organ: it is easy to point at items with gaze while viewing them, but making a selection requires special techniques (Chapter 5 reviewed techniques and methods used in gaze-based text entry). The most commonplace method is to use dwell time, a prolonged gaze on the item under focus. We used dwell time as the selection method in all of our experiments because of its simplicity and because it enables selection by gaze *alone*.

Previous research has shown that text entry by gaze is slow (about 10–15 wpm), but the typing speed can be increased by exploiting character and word prediction methods (summarized in Chapter 6). Since we were interested in studying the very basics of gaze typing, we did not use any prediction in our experiments.

The measured point of gaze is not as accurate an input device as is pointing by hand by means of a conventional mouse. If the user has involuntary eye or head movements, the accuracy achievable in practice can decrease even more. That is why the targets on the screen have to be quite large; sometimes only a few items can be shown at a time. The most common case involves a large on-screen keyboard that shows the full

alphabet. We used the QWERTY layout in most of our studies because of its general familiarity. Since the keyboard occupies most of the screen real estate, there is not much space left for other applications.

Chapter 7 discussed issues related to layout and summarized the results of an experiment with scrollable keyboards. The idea of the scrollable keyboard is to use a layout familiar to the user (such as QWERTY) and to save screen space by showing only part of it. Scrollable keyboards reduce the space taken by the full (three-row) keyboard by 1/3 (if two of the three rows are shown) or 2/3 (if only one row is shown). In the study, typing speed fell by only 51.4% for the one-row and 25.3% for the two-row keyboard, in comparison with the conventional QWERTY layout. Furthermore, the increased keystroke rate was quite reasonable, from 1 KSPC to 1.64 KSPC and 1.2 KSPC with the one-row and two-row keyboard, respectively. By optimizing the keyboard layout according to letter-to-letter probabilities, it was possible to further increase the typing speed. However, since the optimized layout is unfamiliar to the user, it requires a longer learning time and may no longer be immediately usable. We believe scrollable keyboards would be especially useful in casual typing situations where an overview of the application or a Web page is a more important consideration than a slight reduction in typing speed is.

Appropriate feedback is especially important when the same modality is used for input and output. The user's gaze is engaged in the typing process: he or she needs to look at the characters while selecting them. The user cannot simultaneously see the typed text and needs to move the gaze from the keyboard to the text input field in order to review the result. The need to switch between the keyboard and the text field can be reduced by enhancing the feedback so that the user feels confident in the selection without a need to check the result.

Chapter 8 provided a brief review of previous research and presented results from three experiments studying the effects of feedback on gaze typing. The results show that the type of feedback significantly affects typing speed, accuracy, gaze behavior, and subjective experience. For example, visual feedback combined with a short audible "click" significantly facilitates eye typing. Compared with plain visual feedback, added auditory feedback significantly increases typing speed and reduces errors. Spoken feedback can be useful for novices using long dwell times: speaking out the letters as they are typed significantly helps to reduce errors. However, with short dwell times, spoken feedback is problematic since speaking a letter takes time. As our experiment demonstrated, people tend to pause to listen to the speech. This not only decreases the typing speed but also introduces double-entry errors: the same letter is unintentionally typed twice. For novices, it may also be useful to give extra feedback on the dwell time progress. This can be done via animation

or by showing two-level feedback: first, feedback on the focus and, after the dwell time has elapsed, feedback on the selection. This gives the user a chance to cancel the selection before the dwell time runs out. It is not natural to fixate on a target for a long time. Animated feedback helps in maintaining focus on the target.

When a short dwell time is used, there may not be enough time to give extra feedback to the user. With short dwell times, the participants found the two-level feedback confusing and distracting. The feedback should match the dwell time. Short dwell times require simplified feedback, while long dwell times allow extra information on the eye typing process. The same dwell time (e.g., 500 ms) may be “short” for one user and “long” for another. Therefore, the user should be able to adjust the dwell time as well as the feedback parameters and attributes.

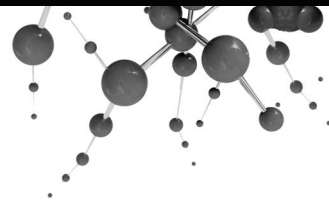
Even though it is natural to point at items by gaze, it takes some time to learn to use gaze as a means for controlling a computer. This is especially true with novel gaze typing methods such as Dasher. Chapter 9 briefly introduced issues related to learning gaze interaction and reported results from two experiments. The first experiment studied how novices learn to write by gaze alone with Dasher. It was noted that, while Dasher is one of the most discussed inventions of recent years in the text entry field and is acknowledged as the world’s fastest method of entering text by gaze, no independent experiments on gaze writing with Dasher had been published to verify the results of Ward and MacKay (2002), despite all the attention. We conducted a longitudinal study in which 12 novice participants transcribed Finnish text with Dasher in 10 15-minute sessions, using a Tobii 1750 eye tracker as a pointing device. The results confirmed that people can enter text quite efficiently with Dasher by using gaze alone. After 2.5 hours of practice, participants were able to enter text at an average rate of 17.3 wpm. This is somewhat slower than we expected, but it is at least partly explained by the quality of the corpus used to build the language model that Dasher used in our experiment. In addition, the learning curve was still growing after the 10 sessions, indicating that the typing speed would have still increased significantly if the experiment had continued.

Gaze typing using dwell time is considered to be slow (especially in comparison to use of Dasher). Therefore, we were also interested in seeing how quickly novices can learn to type by gaze with the most typical setup: an on-screen keyboard and dwell time control. Most gaze-based text entry evaluations have been conducted with novices using a constant, fairly long dwell time. We conducted a longitudinal study to find out how rapidly novices learn to type by gaze when they are allowed to adjust the dwell time at will as they wish. This study with an on-screen QWERTY keyboard followed the method used in the Dasher study, with an equal

amount of practice and similar test procedures. The results showed that the text entry rate increased from 6.9 wpm in the first session to 19.9 wpm in the tenth. Correspondingly, the dwell time decreased from an average of 876 ms to 282 ms. The final typing speed of nearly 20 wpm in this study is comparable with the results in the Dasher study (with an average of 17.3 wpm in the tenth session). We thus conclude that people can gaze type fairly rapidly and accurately when using a simple, easy-to-learn on-screen keyboard, if a fixed dwell time does not slow down the typing.

Gaze-based text entry systems typically provide a backspace or undo key for immediate corrections. However, since the keyboard itself takes a lot of space, the editing commands are often hidden in the virtual keyboard's menu structure. This was the case in our experiments also: participants were able to correct mistakes only by using a backspace key, but there was no way to navigate in the text to correct a mistake in the middle of the sentence. Gaze-based text editing offers a rich set of opportunities for future research and development. As a first step, we developed a dynamic pie-like menu that can potentially facilitate the task of text editing by gaze. We compared the dynamic pie menu with a static editing menu in an initial pilot study with 13 participants. Preliminary results indicate that the dynamic pie menu may be useful, particularly in simple editing tasks. However, further development of the pie menu on the basis of preliminary results and more research is needed before any definitive conclusions are drawn.

In addition to research on editing of text by gaze, other directions for future research include better involvement of users with disabilities, including ways to remotely evaluate usability of the systems and to organize longitudinal studies with the target users.



12 References

- Aoki, H., Hansen, J.P., & Itoh, K. (2006) Towards remote evaluation of gaze typing systems. *Proceedings of the 2nd Conference on Communication by Gaze Interaction (COGAIN 2006)*, 96-103. Available at http://www.cogain.org/cogain2006/COGAIN2006_Proceedings.pdf (accessed 14 February 2009).
- Aoki, H., Hansen, J.P., & Itoh, K. (2008) Learning to interact with a computer by gaze. *Behaviour and Information Technology* 27(4), 339-344.
- Ashmore, M., Duchowski, A.T., & Shoemaker, G. (2005) Efficient eye pointing with a fisheye lens. *Proceedings of Graphics Interface 2005 (GI'05)*, 203-210. Ontario, Canada: Canadian Human-Computer Communications Society (CHCCS).
- Baecker, R., Small, I., & Mander, R. (1991) Bringing icons to life. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '91)*, 1-6. New York: ACM Press.
- Barea, R., Boquete, L., Mazo, M., & Lopez, E. (2002) System for assisted mobility using eye movements based on electro-oculography. *IEEE Transactions on Neural Systems and Rehabilitation Engineering* 10(4), 209-218.
- Barreto, A.B., Scargle, S.D., & Adjouadi, M. (2000) A practical EMG-based human-computer interface for users with motor disabilities. *Journal of Rehabilitation Research and Development* 37(1), 53-64.
- Bates, R. (2002) Have Patience with your eye mouse! Eye-gaze interaction with computers can work. *Proceedings of the 1st Cambridge Workshop on*

- Universal Access and Assistive Technology* (CWUAAT'02), 33-38. Available at <http://www.cse.dmu.ac.uk/~rbates/Bates7.pdf> (accessed 14 February 2009).
- Bates, R., Donegan, M., Istance, H.O., Hansen, J.P., & R ih a, K.-J. (2006) Introducing COGAIN – Communication by Gaze Interaction. In J. Clarkson, P. Langdon & P. Robinson (Eds.) *Designing Accessible Technology, Part II "Enabling Computer Access and the Development of New Technologies"*, 77-84. London: Springer-Verlag.
- Bates, R. & Istance, H.O. (2002) Zooming interfaces! Enhancing the performance of eye controlled pointing devices. *Proceedings of the Fifth international ACM Conference on Assistive Technologies (ASSETS'02)*, 119-126. New York: ACM Press.
- Bates, R. & Istance, H.O. (2003) Why are eye mice unpopular? A detailed comparison of head and eye controlled assistive technology pointing devices. *Universal Access in the Information Society* 2(3), 280-290.
- Bates, R., Istance, H.O., & Vickers, S. (2008) Gaze interaction with virtual on-line communities. *Designing Inclusive Futures*, 149-162. London: Springer.
- Bee, N. & Andr e, E. (2008) Writing with your eye: A dwell time free writing system adapted to the nature of human eye gaze. *Perception in Multimodal Dialogue Systems, LNCS 5078/2008*, 111-122. Springer Berlin/Heidelberg.
- Betke, M., Gips, J., & Fleming, P. (2002) The camera mouse: visual tracking of body features to provide computer access for people with severe disabilities. *IEEE Transactions on Neural Systems and Rehabilitation Engineering* 10(1), 1-10.
- Beukelman, D.R. & Mirenda, P. (1992) *Augmentative and Alternative Communication: Management of Severe Communication Disorders in Children and Adults*. Paul H. Brookes Publishing Co, Baltimore.
- Brewster, S.A. & Crease, M.G. (1999) Correcting menu usability problems with sound. *Behaviour and Information Technology* 18(3), 165-177.
- Brewster, S.A. R aty, V.-P., & Kortekangas, A. (1996) Enhancing scanning input with non-speech sounds. *Proceedings of the Second Annual ACM Conference on Assistive Technologies (ASSETS'96)*, 10-14. New York: ACM Press.
- Bonino, D., Castellina, E., Corno, F., Gale, A., Garbo, A., Purdy, K., & Shi, F. (2009) A blueprint for integrated eye-controlled environments. *Universal Access in the Information Society* 8(4), Springer. (Online First

version published by the time of writing this thesis, DOI:
<http://dx.doi.org/10.1007/s10209-009-0145-4>)

- Bulling, A., Roggen, D., & Tröster, G. (2008a). It's in your eyes - Towards context-awareness and mobile HCI using wearable EOG goggles. *Proceedings of the 10th International Conference on Ubiquitous Computing (UbiComp'08)*, 84-93. New York: ACM Press.
- Bulling, A., Roggen, D., & Tröster, G. (2008b) EyeMote - Towards context-aware gaming using eye movements recorded from wearable electrooculography. *Proceedings of the Second International Conference on Fun and Games, LNCS 5294*, 33-45. Springer Berlin / Heidelberg.
- Calvo, A., Chiò, A., Castellina, E., Corno, F., Farinetti, L., Ghiglione, P., Pasian, V. & Vignola, A. (2008) Eye tracking impact on quality-of-life of ALS patients. *Computers Helping People with Special Needs (ICCHP'08), LNCS 5105/2008*, pp. 70-77. Berlin: Springer.
- Castellina, E. & Corno, F. (2007) Accessible web surfing through gaze interaction. *Proceedings of the 3rd Conference on Communication by Gaze Interaction (COGAIN 2007)*, 74-77. Available at <http://www.cogain.org/cogain2007/COGAIN2007Proceedings.pdf> (accessed 14 February 2009).
- Chapman, J.E. (1991). The use of eye-operated computer system in locked-in syndrome. *Proceedings of the Sixth Annual International Conference on Technology and Persons with Disabilities (CSUN'91)*, Los Angeles, CA.
- Charlier J., Buquet, C., Dubus, F., Hugeux, J.P., & Degroc, B. (1997) VISIOBOARD: A new gaze command system for handicapped subjects. *Medical and Biological Engineering and Computing* 35(416), supplement D90.OS1.03.
- Cleveland, N. (1994) Eyegaze human-computer interface for people with disabilities. *Proceedings of 1st Automation Technology and Human Performance Conference*, Washington DC.
- Corno, F., Gale, A., Majaranta, P., & Rähkä, K.-J. (2009, in press) Eye-based Direct Interaction for Environmental Control in Heterogeneous Smart Environments. To Appear in *Handbook of Ambient Intelligence and Smart Environments*, Springer.
- Dasher Homepage (2008) <http://www.dasher.org.uk> (accessed 4 February 2009).
- Demasco, P.W. & McCoy, K.F. (1992) Generating text from compressed input: An intelligent interface for people with severe motor impairments. *Communications of the ACM* 35(5), 68-78.

- DiMattia, P., Curran, F.X., & Gips, J. (2001) *An Eye Control Teaching Device for Students without Language Expressive Capacity: EagleEyes*. Lampeter, U.K.: Edwin Mellen.
- Donegan, M., Morris, D.J., Corno, F., Signorile, I., Chió, A., Pasian, V., Vignola, A., Buchholz, M., & Holmqvist, E. (2009) Understanding users and their needs. *Universal Access in the Information Society* 8(4), Springer. (Online First version published by the time of writing this thesis, DOI: <http://dx.doi.org/10.1007/s10209-009-0148-1>).
- Donegan, M. & Oosthuizen, L. (2006) The 'KEE' concept for eye-control and complex disabilities: Knowledge-based, End-user focused and Evolutionary. *Proceedings of the 2nd Conference on Communication by Gaze Interaction (COGAIN 2006)*, 83-87. Available at http://www.cogain.org/cogain2006/COGAIN2006_Proceedings.pdf (accessed 14 February 2009).
- Donegan, M., Oosthuizen, L., Bates, R., Daunys, G., Hansen, J.P., Joos, M., et al. (2005) *D3.1 User requirements report with observations of difficulties users are experiencing*. Communication by Gaze Interaction (COGAIN). IST-2003-511598: Deliverable 3.1. Available at <http://www.cogain.org/results/reports/COGAIN-D3.1.pdf> (accessed 14 February 2009).
- Donegan, M. Oosthuizen, L. Daunys, G., Istance, H. Bates, R. Signorile, I., et al. (2006a) *D3.2 Report on features of the different systems and development needs*. Communication by Gaze Interaction (COGAIN). IST-2003-511598: Deliverable 3.2. Available at <http://www.cogain.org/results/reports/COGAIN-D3.2.pdf> (accessed 14 February 2009).
- Donegan, M. Oosthuizen, L., Bates, R., Istance, H., Holmqvist, E., Lundälv, M., et al. (2006b) *D3.3 Report of User Trials and Usability Studies*. Communication by Gaze Interaction (COGAIN). IST-2003-511598: Deliverable 3.3. Available at <http://www.cogain.org/results/reports/COGAIN-D3.3.pdf> (accessed 14 February 2009).
- Dorr, M., Böhme, M., Martinetz, T., & Barth, E. (2007) Gaze beats mouse: a case study. *Proceedings of the 3rd Conference on Communication by Gaze Interaction (COGAIN 2007)*, 16-19. Available at <http://www.cogain.org/cogain2007/COGAIN2007Proceedings.pdf> (accessed 14 February 2009).
- Drewes, H. & Schmidt, A. (2007) Interacting with the computer using gaze gestures. *Proceedings of INTERACT '07, LNCS 4663*, 475-488. Springer.

- Duchowski, A.T. (2003) *Eye Tracking Methodology: Theory and Practice*. London: Springer-Verlag.
- Duchowski, A. T. & Vertegaal, R. (2000) *Eye-Based Interaction in Graphical Systems: Theory and Practice*. Course 05, SIGGRAPH 2000. Course Notes. New York: ACM Press. Course notes are available at <http://vret.ces.clemson.edu/sigcourse/> (accessed 14 February 2009).
- EagleEyes (2000). *EagleEyes for Windows: User Manual*. Boston College, MA, USA. Available at <http://www.cs.bc.edu/~eagleeye/manuals.html> (accessed 14 February 2009).
- Evreinov, G.E. & Raisamo, R. (2004) Optimizing menu selection process for single-switch manipulation. *Proceedings of the 9th International Conference on Computers Helping People with Special Needs (ICCHP'04)*, LNCS 3118/2004, 836-844. Springer Berlin / Heidelberg.
- EyeTech (2005) *Quick Glance 2 User's Guide*. EyeTech Digital Systems, Inc.
- Fejtová, M., Fejt, J., & Lhotská, L. (2004) Controlling a PC by eye movements: The MEMREC project. *Proceedings of the 9th International Conference on Computers Helping People with Special Needs (ICCHP '04)*, LNCS 3118/2004, 770-773. Springer Berlin / Heidelberg.
- Fejtová, M., Novák, P., Fejt, J., & Štěpánková, O. (2006) When can eyes make up for hands. *Proceedings of the 2nd Conference on Communication by Gaze Interaction (COGAIN 2006)*, 46-49. Available at http://www.cogain.org/cogain2006/COGAIN2006_Proceedings.pdf (accessed 14 February 2009).
- Fono, D. & Vertegaal, R. (2005) EyeWindows: Evaluation of eye-controlled zooming windows for focus selection. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI'05)*, 151-160. New York: ACM Press.
- Frey, L.A, White, K.P. Jr., & Hutchinson, T.E. (1990) Eye-gaze word processing. *IEEE Transactions on Systems, Man, and Cybernetics* 20(4), 944-950.
- Friedman, M.B., Kiliany, G., Dzmura, M. & Anderson, D. (1982) The eyetracker communication system. *Johns Hopkins APL Technical Digest* 3(3), 250-252.
- Garbe, J. (2006) *Typing Quickly and Relaxed with the Eyes. A case study comparing switch based and Gaze Controlled Input Methods*. Handout for the COGAIN PhD Course on Eye-Computer Interaction: Eye Performance and Interface Design, 6-8 September 2006, Turin, Italy.

Available at http://www.cogain.org/events/camp2006/phd_course/garbe/Garbe-Handout.pdf (accessed 14 February 2009).

- Gaver, W.W. (1989) The SonicFinder: An interface that uses auditory icons. *Human-Computer Interaction* 4(1), 67-94.
- Gips, J., DiMattia, P., Curran, F.X., & Olivieri, P. (1996) Using EagleEyes – An electrodes based device for controlling the computer with your eyes – to help people with special needs. In J. Klaus, E. Auff, W. Kremser and W. Zagler (Eds), *Interdisciplinary Aspects on Computers Helping People with Special Needs – Proceedings of the 5th International Conference on Computers Helping People with Special Needs (ICCHP '96)*, 630-635. Vienna: R. Oldenburg.
- Gips, J. & Olivieri, P. (1996) EagleEyes: An Eye Control System for Persons with Disabilities. Presented at *The Eleventh International Conference on Technology and Persons with Disabilities*, Los Angeles, CA. Available at <http://www.cs.bc.edu/~eagleeye/papers/paper1/paper1.html> (accessed 14 February 2009).
- Gips, J., Olivieri, C.P., & Tecce, J.J. (1993) Direct control of the computer through electrodes placed around the eyes. In M. J. Smith and G. Salvendy (Eds.), *Human-computer interaction: Applications and case studies (Proceedings of HCI International '93)*, 630-635. Amsterdam: Elsevier.
- Glenstrup A.J. & Engell-Nielsen, T. (1995) *Eye controlled media: Present and future of state*. Technical report, University of Copenhagen. Available at <http://www.diku.dk/~panic/eyegaze/> (accessed 14 February 2009).
- Goldberg, J.H. & Wichansky, A.M. (2003) Eye tracking in usability evaluation: A practitioner's guide. In J. Hyönä, R. Radach and H. Deubel (Eds.), *The mind's eye: Cognitive and applied aspects of eye movement research*, 493-516. Amsterdam, The Netherlands: North-Holland.
- Goossens', C.A. & Crain, S.S. (1987) Overview of nonelectronic eye gaze communication techniques. *Augmentative and Alternative Communication* 3, 77-89.
- Grauman, K., Betke, M., Lombardi, J., Gips, J., & Bradski, G.R. (2003) Communication via eye blinks and eyebrow raises: Video-based human-computer interfaces. *Universal Access in the Information Society* 2(4), 359-373.
- Haber, R.N. & Hershenson, M. (1973) *The Psychology of Visual Perception*. London: Holt, Rinehart and Winston.

- Hansen, D.W. & Hansen, J.P. (2006). Eye typing with common cameras. *Proceedings of the Symposium on Eye Tracking Research & Applications (ETRA'06)*, 55. New York: ACM Press.
- Hansen, D.W., Hansen, J.P., Nielsen, M., Johansen, A.S., & Stegmann, M.B. (2002) Eye typing using Markov and active appearance models. *Proceedings of the Sixth IEEE Workshop on Applications of Computer Vision (WACV'02)*, 132-136. IEEE Computer Society.
- Hansen, D.W. & Ji, Q. (2009) In the Eye of the beholder: A survey of models for eyes and gaze. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23 Jan. 2009. IEEE computer Society Digital Library. IEEE Computer Society.
- Hansen, D.W., Skovsgaard, H.H., Hansen, J.P., & Møllenbach, E. (2008) Noise tolerant selection by gaze-controlled pan and zoom in 3D. *Proceedings of the Symposium on Eye Tracking Research & Applications (ETRA'08)*, 205-212. New York: ACM Press.
- Hansen, J.P., Andersen, A.W., & Roed, P. (1995) Eye-gaze control of multimedia systems. In Y. Anzai, K. Ogawa and H. Mori (Eds.) *Symbiosis of Human and Artifact - Proceedings of the 6th International Conference on Human Computer Interaction (HCII '95)*, 37-42. Amsterdam: Elsevier.
- Hansen, J.P., Hansen, D.W., & Johansen, A.S. (2001) Bringing gaze-based interaction back to basics. In C. Stephanidis (Ed.) *Universal Access in HCI (UAHCI): Towards an Information Society for All - Proceedings of the 9th International Conference on Human-Computer Interaction (HCII'01)*, 325-328. Mahwah, NJ: Lawrence Erlbaum Associates.
- Hansen, J.P., Johansen, A.S., Hansen, D.W., Itoh, K., & Mashino, S. (2003a) Command without a click: Dwell time typing by mouse and gaze selections. In M. Rauterberg, M. Menozzi and J. Wesson (Eds.) *Proceedings of 9th IFIP TC13 International Conference on Human-Computer Interaction (INTERACT'03)*, 121-128. Amsterdam: IOS Press.
- Hansen, J.P., Johansen, A.S., Hansen, D.W., Itoh, K., & Mashino, S. (2003b) Language technology in a predictive, restricted on-screen keyboard with ambiguous layout for severely disabled people. *Workshop on Language Modeling for Text Entry Methods (EACL'03)*, Budapest, Hungary. Available at http://www.it-c.dk/research/EyeGazeInteraction/Papers/Hansen_et_al_2003a.pdf (accessed 14 February 2009).
- Hansen, J.P., Tørning, K., Johansen, A.S., Itoh, K., & Aoki, H. (2004) Gaze typing compared with input by head and hand. *Proceedings of the*

Symposium on Eye Tracking Research & Applications (ETRA'04), 131-138. New York: ACM Press.

Harbusch, K. & Kühn, M. (2003) Towards an adaptive communication aid with text input from ambiguous keyboards. *Proceedings of the Tenth Conference on European Chapter of the Association For Computational Linguistics - Volume 2*. European Chapter Meeting of the ACL. Association for Computational Linguistics, Morristown, NJ, 207-210.

Heikkilä, H. (2008) Gesturing with Gaze. *Proceedings of the 4th Conference on Communication by Gaze Interaction (COGAIN 2008): Communication, Environment and Mobility Control by Gaze*, 43-46. Prague: CTU Publishing House, Prague (ISBN 978-80-01-04151-2). Available at <http://www.cogain.org/cogain2008/COGAIN2008-Proceedings.pdf> (accessed 14 February 2009).

Hillstrom, A.P. & Yantis, S. (1994) Visual motion and attentional capture. *Perception & Psychophysics* 55(4), 399-411.

Hori, J., Sakano, K., & Saitoh, Y. (2006) Development of a communication support device controlled by eye movements and voluntary eye blink. *IEICE transactions on information and systems* 89(6), 1790-1797.

Hornof, A. (2008) Working with children with severe motor impairments as design partners. *Proceedings of the 7th International Conference on interaction Design and Children (IDC'08)*, 69-72. New York: ACM Press.

Hornof, A.J. & Cavender, A. (2005) EyeDraw: Enabling children with severe motor impairments to draw with their eyes. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI'05)*, 161-170. New York: ACM Press.

Hornof, A., Cavender, A., & Hoselton, R. (2004) EyeDraw: A System for Drawing Pictures with Eye Movements. *Proceedings of the 6th International ACM SIGACCESS Conference on Computers and Accessibility (ASSETS'04)*, 86-93. New York: ACM Press.

Huckauf, A. (2005) Controlling computers by eyes while reducing Midas touch-problems. In M. Groner, R. Groner, R. Müri, K. Koga, S. Raess & P. Sury (Eds.) *Journal of Eye Movement Research Special Issue: Abstracts of the Thirteenth European Conference on Eye Movements (ECEM13)*, PB-184 (p. 111). Available at <http://www.jemr.org/online/1/s2> (accessed 14 February 2009).

Huckauf, A., Goettel, T., Heinbockel, M., & Urbina, M.H. (2005) What you don't look at is what you get: anti-saccades can reduce the Midas touch-problem. *Proceedings of the 2nd Symposium on Applied Perception in Graphics and Visualization (APGV'05)*, 170. New York: ACM Press.

- Huckauf, A. & Urbina, M. (2007) Gazing with pEYE: new concepts in eye typing. *Proceedings of the 4th Symposium on Applied Perception in Graphics and Visualization (APGV'07)*, 141-141 New York: ACM Press.
- Huckauf, A. & Urbina, M.H. (2008a) Gazing with pEYEs: Towards a universal input for various applications. *Proceedings of the Symposium on Eyetracking Research & Applications (ETRA'08)*, 51-54. New York: ACM Press.
- Huckauf, A. & Urbina, M.H. (2008b). On object selection in gaze controlled environments. *Journal of Eye Movement Research* 2(4), 4, 1-7.
- Hutchinson, T.E., White, K.P., Martin, W.N., Reichert, K.C., & Frey, L.A. (1989) Human-computer interaction using eye-gaze input. *IEEE Transactions on Systems, Man, and Cybernetics* 19(6), 1527-1534.
- Hyrskykari, A., Majaranta, P., & Rähkä, K.-J. (2003) Proactive Response to Eye Movements. In M. Rauterberg, M. Menozzi, and J. Wesson (Eds.), *Proceedings of 9th IFIP TC13 International Conference on Human-Computer Interaction (INTERACT'03)*, 129-136. Amsterdam: IOS Press.
- Hyrskykari, A., Majaranta, P., & Rähkä, K.-J. (2005) From gaze control to attentive interfaces. In C. Stephanidis (Ed.) *Universal Access in HCI: Exploring New Interaction Environments - Volume 7 of the Proceedings of the 11th International Conference on Human-Computer Interaction (HCII'05)*, (CD-ROM). Mahwah, NJ: Lawrence Erlbaum Associates, Inc.
- Isokoski, P. (2000) Text input methods for eye trackers using off-screen targets. *Proceedings of the Symposium on Eye Tracking Research & Applications (ETRA'00)*, 15-21. New York: ACM Press.
- Isokoski, P. & Linden, T. (2004) Effect of foreign language on text transcription performance: Finns writing English. *Proceedings of the third Nordic conference on Human-computer interaction (NordiCHI'04)*, 105-108. New York: ACM Press.
- Isokoski, P. & Martin, B. (2006) Eye tracker input in first person shooter games. *Proceedings of the 2nd Conference on Communication by Gaze Interaction (COGAIN 2006)*, 76-79. Available at http://www.cogain.org/cogain2006/COGAIN2006_Proceedings.pdf (accessed 14 February 2009).
- Isokoski, P. & Raisamo, R. (2000) Device independent text input: A rationale and an example. *Proceedings of the Working Conference on Advanced Visual interfaces (AVI '00)*, 76-83. New York: ACM Press.

- Isokoski, P. & Raisamo, R. (2004) Quikwriting as a multi-device text entry method. *Proceedings of the 3rd Nordic Conference on Human-Computer Interaction (NordiCHI '04)*, 109-112. New York: ACM Press.
- Isokoski, P., Joos, M., Spakov, O. & Martin, B. (2009) Gaze controlled games. *Universal Access in the Information Society*, 8(4), Springer. (Online First version published by the time of writing this thesis, DOI: <http://dx.doi.org/10.1007/s10209-009-0146-3>)
- Istance, H.O., Bates, R., Hyrskykari, A., & Vickers, S. (2008) Snap clutch, a moded approach to solving the Midas touch problem. *Proceedings of the Symposium on Eye Tracking Research & Applications (ETRA'08)*, 221-228. New York: ACM Press.
- Istance, H.O., Spinner, C., & Howarth, P.A. (1996) Providing motor impaired users with access to standard Graphical User Interface (GUI) software via eye-based interaction. *Proceedings of the 1st European Conference on Disability, Virtual Reality and Associated Technologies (ECDVRAT'96)*, 109-116. Available at: http://www.icdvrat.reading.ac.uk/1996/papers/1996_13.pdf (accessed 14 February 2009).
- Itoh, K., Aoki, H., & Hansen, J.P. (2006) A comparative usability study of two Japanese gaze typing systems. *Proceedings of the Symposium on Eye Tracking Research & Applications (ETRA'06)*, 59-66. New York: ACM.
- Jacob, R.J.K. (1991) The use of eye movements in human-computer interaction techniques: what you look at is what you get. *ACM Transactions on Information Systems* 9(3), 152-169.
- Jacob, R.J.K. (1993) Eye movement-based human-computer interaction techniques: Toward non-command interfaces. In H. R. Hartson and D. Hix (Eds.) *Advances in Human-Computer Interaction*, Vol. 4, 151-190. Ablex Publishing Co.
- Jacob, R.J.K. (1995) Eye tracking in advanced interface design. In W. Barfield and T. A. Furness (Eds.) *Virtual Environments and Advanced Interface Design*, 258-288. New York: Oxford University Press.
- Jacob, R.J.K. & Karn, K.S. (2003) Eye tracking in human-computer interaction and usability research: Ready to deliver the promises (section commentary). In J. Hyönä, R. Radach, and H. Deubel (Eds.) *The Mind's Eye: Cognitive and Applied Aspects of Eye Movement Research*, 573-605. Amsterdam: Elsevier Science.
- Joos, M., Malischke, S., Pannasch, S., Storch, A., & Velichkovsky, B.M. (2007) Comparing two gaze-interaction interfaces: A usability study with locked-in patients. *Proceedings of the 3rd Conference on Communication by Gaze Interaction (COGAIN 2007)*, 82-88. Available at

<http://www.cogain.org/cogain2007/COGAIN2007Proceedings.pdf>
(accessed 14 February 2009).

Jordansen, I.K., Boedeker, S., Donegan, M., Oosthuizen, L., di Girolamo, M., & Hansen, J.P. (2005) *D7.2 Report on a market study and demographics of user population*. Communication by Gaze Interaction (COGAIN). IST-2003-511598: Deliverable 7.2. Available at <http://www.cogain.org/results/reports/COGAIN-D7.2.pdf> (accessed 14 February 2009).

Junker, A.M. & Hansen, J.P. (2006) Gaze pointing and facial EMG clicking. *Proceedings of the 2nd Conference on Communication by Gaze Interaction (COGAIN 2006)*, 83-87. Available at http://www.cogain.org/cogain2006/COGAIN2006_Proceedings.pdf (accessed 14 February 2009).

Kahn, D.A., Heynen, J., & Snuggs, G.L. (1999) Eye-controlled computing: The VisionKey experience. *Proceedings of the Fourteenth International Conference on Technology and Persons with Disabilities (CSUN'99)*. Los Angeles, CA.

Kammerer, Y., Scheiter, K., and Beinhauer, W. (2008) Looking my way through the menu: The impact of menu design and multimodal input on gaze-based menu selection. *Proceedings of the Eye Tracking Research & Applications Symposium (ETRA'08)*, 213-220. New York: ACM Press.

Kaur, M., Tremaine, M., Huang, N., Wilder, J., Gacovski, Z., Flippo, F., & Mantravadi, C. S. (2003) Where is "it"? Event synchronization in gaze-speech input systems. *Proceedings of the 5th international conference on Multimodal interfaces (ICMI'03)*, 151-158. New York: ACM Press.

Koester, H.H. & Levine, S.P. (1994a) Learning and performance of able-bodied individuals using scanning systems with and without word prediction. *Assistive Technology* 6(1), 42-53.

Koester, H.H. & Levine, S.P. (1994b). Modeling the speed of text entry with a word prediction interface. *IEEE Transactions on Rehabilitation Engineering* 2(3), 177-187.

Kristensson, P. & Zhai, S. (2004) SHARK2: a large vocabulary shorthand writing system for pen-based computers. *Proceedings of the 17th Annual ACM Symposium on User interface Software and Technology (UIST'04)*, 43-52. New York: ACM Press.

Kumar, M., Paepcke, A., & Winograd, T. (2007) EyePoint: Practical pointing and selection using gaze and keyboard. *Proceedings of SIGCHI Conference on Human Factors in Computing Systems (CHI'07)*, 421-430. New York: ACM Press.

- Kurtenbach, G. & Buxton, W. (1994) User learning and performance with marking menus. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI'94)*, 258-264. New York: ACM Press.
- Lankford, C. (2000) Effective eye-gaze input into Windows. *Proceedings of the Symposium on Eye Tracking Research and Applications (ETRA'00)*, 23-27. New York: ACM Press.
- Lepistö, A. & Ovaska, S. (2004) Usability evaluation involving participants with cognitive disabilities. *Proceedings of the Third Nordic Conference on Human-Computer interaction (NordiCHI'04)*, vol. 82, 305-308. New York: ACM Press.
- Levine, J.L. (1981) *An Eye-Controlled Computer*. Research report RC-8857, IBM Thomas J. Watson Research Center, Yorktown Heights, N.Y.
- Lund, H. & Hansen, J.P. (2008) Gaze interaction and access to library collection. *Research and Advanced Technology for Digital Libraries, LNCS 5173*, 423-424. Springer Berlin/Heidelberg.
- MacDonald, A. (1998) Symbol systems. In Allan Wilson (Ed.) *Augmentative Communication in Practice: An Introduction* (2nd ed.), 19-26.
- MacKay, D.J.C. (2006) Dasher Manual. Available at <http://www.inference.phy.cam.ac.uk/dasher/download/papers/Manual.pdf> (accessed 14 February 2009).
- MacKenzie, I.S. (2002) KSPC (keystrokes per character) as a characteristic of text entry techniques. *Proceedings of the Fourth International Symposium on Human-Computer Interaction with Mobile Devices (MobileHCI'02)*, 195-210. Heidelberg: Springer-Verlag.
- MacKenzie, I.S. (2003) Motor behaviour models for human-computer interaction. In J. M. Carroll (Ed.), *Toward a Multidisciplinary Science of Human-Computer Interaction*, 27-54. Morgan Kaufmann.
- MacKenzie, I.S., Chen, J., & Oniszczak, A. (2006) Unipad: Single-stroke text entry with language-based acceleration. *Proceedings of the Fourth Nordic Conference on Human-Computer Interaction (NordiCHI'06)*, 78-85. New York: ACM.
- MacKenzie, I.S. & Soukoreff, R.W. (2003) Phrase sets for evaluating text entry techniques. *Extended Abstracts on Human Factors in Computing Systems (CHI'03)*, 754-755. New York: ACM Press.
- MacKenzie, I.S. & Tanaka-Ishii, K. (2007) *Text Entry Systems: Mobility, Accessibility, Universality*. San Francisco: Morgan Kaufmann.

- MacKenzie, I.S., Zhang, S.X., & Soukoreff, R.W. (1999) Text entry using soft keyboards. *Behaviour & Information Technology* 18, 235-244.
- MacKenzie, I.S. & Zhang, X. (2008) Eye typing using word and letter prediction and a fixation algorithm. *Proceedings of the Symposium on Eye Tracking Research & Applications (ETRA'08)*, 55-58. New York: ACM Press (DOI: 10.1145/1344471.1344484).
- Majaranta, P., Ahola, U.-K., & Špakov, O. (2009b) Fast gaze typing with an adjustable dwell time. *Proceedings of the 27th International Conference on Human Factors in Computing Systems (CHI'09)*, 357-360. New York: ACM Press (DOI: 10.1145/1518701.1518758).
- Majaranta, P., Aula, A., & Rähä, K.-J. (2004) Effects of feedback on eye typing with a short dwell time. *Proceedings of the Symposium on Eye Tracking Research & Applications (ETRA'04)*, 139-146. New York: ACM Press (DOI: 10.1145/968363.968390).
- Majaranta, P., Bates, R., & Donegan, M. (2009a) Eye-tracking. In Constantine Stephanidis (Ed.) *The Universal Access Handbook*, 587-606. Human Factors and Ergonomics series, Lawrence Erlbaum Associates, Inc.
- Majaranta, P., MacKenzie, I. S., Aula, A., & Rähä, K.-J. (2003a) Auditory and visual feedback during eye typing. *Extended Abstracts of the ACM Conference on Human Factors in Computing Systems (CHI'03)*, 766-767. New York: ACM Press (DOI: 10.1145/765891.765979).
- Majaranta, P., MacKenzie, I.S., Aula, A., & Rähä, K.-J. (2006) Effects of feedback and dwell time on eye typing speed and accuracy. *Universal Access in the Information Society* 5(2), 199-208.
- Majaranta, P., MacKenzie, I.S., & Rähä, K.-J. (2003b) Using motion to guide the focus of gaze during eye typing. *Abstracts of the 12th European Conference on Eye Movements (ECM12)*, University of Dundee, O42.
- Majaranta P., Majaranta, N., Daunys, G., & Špakov, O. (2009c) Text editing by gaze. *Proceedings of the 5th Conference on Communication by Gaze Interaction (COGAIN 2009)*, 19-23. IMM-Technical Report, Technical University of Denmark (ISBN 978-87-643-0475-6). Available at <http://www.cogain.org/cogain2009/COGAIN2009-Proceedings.pdf> (accessed 1 June 2009).
- Majaranta, P. & Rähä, K.-J. (2002) Twenty years of eye typing: Systems and design issues. *Proceedings of the Symposium on Eye Tracking Research and Applications (ETRA'02)*, 15-22. New York: ACM (DOI: 10.1145/507072.507076).

- Majaranta, P. & Rähkä, K.-J. (2007) Text entry by gaze: Utilizing eye-tracking. In I. S. MacKenzie and K. Tanaka-Ishii (Eds.) *Text Entry Systems: Mobility, Accessibility, Universality*, 175-187. San Francisco: Morgan Kaufmann.
- Mankoff, J. & Abowd, G.D. (1998) Cirrin: a word-level unistroke keyboard for pen input. *Proceedings of the Symposium on User Interface Software and Technology (UIST'98)*, 213-214. New York: ACM Press.
- Microsoft Windows User Experience Guidelines (2002) Official Guidelines for User Interface Developers and Designers. Microsoft Corporation.
- Milekic, S. (2003) The more you look the more you get: Intention-based interface using gaze-tracking. In *Museums and the Web 2002 Selected Papers from an International Conference*, Archives & Museum Informatics, D. Bearman, and Trant, J. (eds.), Pittsburgh, PA.
- Miniotas, D, Spakov, O., & Evreinov, G.E. (2003) Symbol Creator: An alternative eye-based text entry technique with low demand for screen space. In M. Rauterberg, M. Menozzi, and J. Wesson (Eds.), *Human-Computer Interaction - Proceedings of the IFIP TC13 International Conference on Human-Computer Interaction (INTERACT'03)*, 137-143. IOS Press.
- Miniotas, D., Špakov, O., Tugoy, I., & MacKenzie, I. S. (2006) Speech-augmented eye gaze interaction with small closely spaced targets. *Proceedings of the Symposium on Eye Tracking Research & Applications (ETRA'06)*, 67-72. New York: ACM Press.
- Monden, A., Matsumoto, K., & Yamato, M. (2005) Evaluation of gaze-added target selection methods suitable for general GUIs. *International Journal of Computer Applications in Technology* 24(1), 17-24.
- Morimoto, C.H. & Mimica, M.R.M. (2005) Eye gaze tracking techniques for interactive applications. *Computer Vision and Image Understanding* 98(1), 4-24.
- Murphy, R.A. & Basili, A. (1993) Developing the user-system interface for a communications system for ALS patients and others with severe neurological impairments. Designing for Diversity. *Proceedings of the Human Factors and Ergonomics Society Annual Meeting (HFES'93)*, 2, 854-858. Human Factors and Ergonomics Society.
- Müller-Tomfelde, C. (2007) Dwell-based pointing in applications of human computer interaction. In C. Baranauskas et al. (Eds.), *Proceedings of INTERACT'07, LNCS 4662, Part I*, 560-573.

- Nakano, Y., Nakamura, A., & Kuno, Y. (2004) Web browser controlled by eye movements. *Proceedings of the IASTED International Conference on Advances in Computer Science and Technology (ACST'04)*, 93-98.
- Nielsen, J. (1993) Noncommand user interfaces. *Communications of the ACM* 36(4), 82-99.
- Nielsen, J. & Mack, R.L. (1994) *Usability inspection methods*. New York: John Wiley & Sons.
- Nisbet, P. & Poon, P. (1998) *Special Access Technology*, University of Edinburgh. Available at http://www.callcentrescotland.org.uk/About_CALL/Publications_CAA/Books_CAB/SAT_CAC/sat_cac.html (accessed 14 February 2009).
- Novák, P., Krajník, T., Přeučil, L., Fejtová, M., & Štěpánková, O. (2008) AI support for a gaze controlled wheelchair. *Proceedings of the 4th Conference on Communication by Gaze Interaction (COGAIN 2008): Communication, Environment and Mobility Control by Gaze*, 19-22. Prague: CTU Publishing House, Prague (ISBN 978-80-01-04151-2). Available at <http://www.cogain.org/cogain2008/COGAIN2008-Proceedings.pdf> (accessed 14 February 2009).
- Ohno, T. (1998) Features of eye gaze interface for selection tasks. *Proceedings of the 3rd Asia Pacific Computer-Human Interaction (APCHI'98)*, 176-182. Washington, DC: IEEE Computer Society.
- Porta, M. & Turina, M. (2008) Eye-S: a full-screen input modality for pure eye-based communication. *Proceedings of the Symposium on Eye Tracking Research and Applications (ETRA '08)*, 27-34. New York: ACM Press.
- Perlin, K. (1998) Quikwriting: continuous stylus-based text entry. *Proceedings of the Symposium on User Interface Software and Technology (UIST'98)*, 215-216. New York: ACM Press.
- Quintero, A. (2009) 'Eye' on technology update. Eyegaze users share their experiences. *MDA/ALS Newsmagazine*, 14(3), March 2009. MDA Publications. Available online at http://www.als-mda.org/publications/als/als14_3.html#eye (accessed 6 March 2009).
- Rasmusson, D., Chappell, R., & Trego, M. (1999) Quick Glance: Eye-tracking access to the Windows95 operating environment. *Proceedings of the Fourteenth International Conference on Technology and Persons with Disabilities (CSUN'99)*. Los Angeles, CA.
- Rayner, K. (1995) Eye movements and cognitive processes in reading, visual search, and scene perception. In J. M. Findlay, R. Walker and R.

- W. Kentridge (Eds.) *Eye Movement Research: Mechanisms, Processes and Applications*, 3-22. Amsterdam: North Holland.
- Robertson, C.G., Mackinlay, J.D., & Card, S.K. (1991) Cone trees: Animated 3D visualizations of hierarchical information. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI'91)*, 189-194. New York: ACM Press.
- Salvucci, D.D. (1999) Inferring intent in eye-based interfaces: tracing eye movements with process models. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems: the CHI Is the Limit (CHI '99)*, 254-261. New York: ACM Press.
- Scott, J. (1998) Low tech methods of augmentative communication. In Allan Wilson (Ed.) *Augmentative Communication in Practice: An Introduction* (2nd ed.), 13-18.
- Seifert, K. (2002) *Evaluation Multimodaler Computer-Systeme in Frühen Entwicklungsphasen*. (in German) PhD thesis, Department of Human-Machine Systems, Technical University Berlin. Available at http://edocs.tu-berlin.de/diss/2002/seifert_katharina.pdf. Summary of results involving gaze interaction (in English) available at <http://www.roetting.de/eyes-tea/history/021017/seifert.html> (accessed 14 February 2009).
- Shein, G.F. (1997) *Towards Task Transparency in Alternative Computer Access: Selection of Text Through Switch-Based Scanning*. Ph.D. Thesis, Dept. of Industrial Engineering, University of Toronto.
- Shell, J.S., Vertegaal, R., & Skaburskis, A.W. (2003) EyePliances: attention-seeking devices that respond to visual attention. *Extended Abstracts of Human Factors in Computing Systems (CHI'03)*, 770-771. New York: ACM Press.
- Sibert, L.E. & Jacob, R.J.K. (2000) Evaluation of eye gaze interaction. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '00)*, 281-288. New York: ACM Press.
- Simpson, R. & Koester, H. (1999) Adaptive one-switch row-column scanning. *IEEE Transactions on Rehabilitation Engineering* 7(4), 464-473.
- Skovsgaard, H.H.T., Hansen, J.P., & Mateo, J.C. (2008) How can tiny buttons be hit using gaze only? *Proceedings of the 4th Conference on Communication by Gaze Interaction (COGAIN 2008): Communication, Environment and Mobility Control by Gaze*, 38-42. Prague: CTU Publishing House (ISBN 978-80-01-04151-2). Available at <http://www.cogain.org/cogain2008/COGAIN2008-Proceedings.pdf> (accessed 14 February 2009).

- Smith, J.D. & Graham, T.C. (2006) Use of eye movements for video game control. *Proceedings of the SIGCHI International Conference on Advances in Computer Entertainment Technology (ACE'06)*, article no. 20. New York: ACM Press.
- Soukoreff, R.W. & MacKenzie, I.S. (2001) Measuring errors in text entry tasks: An application of the Levenshtein string distance statistic. *Extended Abstracts on Human Factors in Computing Systems (CHI'01)*, 319-320. New York: ACM Press.
- Soukoreff, R.W. & MacKenzie, I.S. (2003) Metrics for text entry research: An evaluation of MSD and KSPC, and a new unified error metric. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '03)*, 113-120. New York: ACM Press.
- Špakov, O. & Majaranta, P. (2008) Scrollable keyboards for eye typing. *Proceedings of the 4th Conference on Communication by Gaze Interaction (COGAIN 2008)*, 63-66. Prague: CTU Publishing House (ISBN 978-80-01-04151-2). Available at <http://www.cogain.org/cogain2008/COGAIN2008-Proceedings.pdf> (accessed 14 February 2009).
- Špakov, O. & Majaranta, P. (2009, in press) Scrollable keyboards for casual eye typing. To appear in *PsychNology Journal*, in a special issue for Gaze Control for Work and Play.
- Špakov, O. & Miniotas, D. (2004) On-line adjustment of dwell time for target selection by gaze. *Proceedings of the 3rd Nordic Conference on Human-Computer Interaction (NordiCHI'04)*, 203-206. New York: ACM Press.
- Stampe, D.M. & Reingold, E.M. (1995) Selection by looking: A novel computer interface and its application to psychological research. In J. M. Findlay, R. Walker and R. W. Kentridge (Eds.) *Eye Movement Research: Mechanisms, Processes and Applications*, 467-478. Amsterdam: Elsevier Science.
- Surakka, V., Illi, M., & Isokoski, P. (2003) Voluntary eye movements in human-computer interaction. In J. Hyönä, R. Radach, & H. Deubel (Eds.), *The Mind's Eye: Cognitive and Applied Aspects of Eye Movement Research*, 473-491. Amsterdam: Elsevier Science.
- Surakka, V., Illi, M., & Isokoski, P. (2004) Gazing and frowning as a new technique for human-computer interaction. *ACM Transactions on Applied Perception* 1(1), 40-56.
- Tchalenko, J. (2001) Free-eye drawing. *Point: Art and Design Research Journal* 11, 36-41.

- Ten Kate, J.H., Frietman, E.E.E., Willems, W., Ter Haar Romeny, B.M., & Tenkink, E. (1979) Eye-switch controlled communication aids. *Proceedings of the 12th International Conference on Medical and Biological Engineering*, Jerusalem, Israel.
- Tien, G. & Atkins, M. S. (2008) Improving hands-free menu selection using eyegaze glances and fixations. *Proceedings of the Symposium on Eye Tracking Research & Applications Symposium (ETRA'08)*, 47-50. New York: ACM Press.
- Tobii (2006) *User Manual: Tobii Eye Tracker and ClearView analysis software*. Tobii Technology AB.
- Trnka, K., McCaw, J. Yarrington, D., McCoy, K.F., & Pennington, C. (2008) Word prediction and communication rate in AAC. *Proceedings of the 4th IASTED International Conference on Telehealth and Assistive Technologies (Telehealth/AT'08)*, 19-24.
- Tuisku, O., Majaranta, P., Isokoski, P., & Rähkä, K.-J. (2008) Now Dasher! Dash Away! Longitudinal study of fast text entry by eye gaze. *Proceedings of the Symposium on Eye Tracking Research & Applications (ETRA'08)*, 19-26. New York: ACM Press (DOI: 10.1145/1344471.1344476).
- Urbina, M.H. & Huckauf, A. (2007) Dwell time free eye typing approaches. *Proceedings of the 3rd Conference on Communication by Gaze Interaction (COGAIN 2007)*, 65-70. Available at <http://www.cogain.org/cogain2007/COGAIN2007Proceedings.pdf> (accessed 14 February 2008).
- Velichkovsky, B.M. & Hansen, J.P. (1996) New technological windows into mind: there is more in eyes and brains for human-computer interaction. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems: Common Ground (CHI'96)*, 496-503. New York: ACM Press.
- Velichkovsky, B., Sprenger, A., & Unema, P. (1997) Towards gaze-mediated interaction: Collecting solutions of the "Midas touch problem". *Proceedings of the IFIP TC13 International Conference on Human-Computer Interaction (INTERACT'97)*, 509-516. London: Chapman and Hall.
- Vertegaal, R. (2003) Attentive user interfaces. *Communications of the ACM* 46(3), 30-33.
- Vesterby, T. Voss, J.C., Hansen, J.P., Glenstrup, A.J., Hansen, D.W., & Rudolph, M. (2005) Gaze-guided viewing of interactive movies. *Digital Creativity* 16(4), 193-204.

- Vickers, S., Bates, R., & Istance, H. (2008) Gazing into a second life: Gaze-driven adventures, control barriers, and the need for disability privacy in an online virtual world. *Proceedings of the 7th International Conference on Disability, Virtual Reality and Associated Technologies (ICDVRAT'08)*, Maia, Portugal, 8th-10th September 2008. Available at <http://www.icdvrat.reading.ac.uk/2008/index.htm> (accessed 14 February 2008).
- Wade, N.J. & Tatler, B.W. (2005) *The Moving Tablet of the Eye: The Origins of Modern Eye Movement Research*. Oxford: Oxford University Press.
- Ward, D.J., Blackwell, A.F., & MacKay, D.J.C. (2000) Dasher – a data entry interface using continuous gestures and language models. *Proceedings of the 13th Annual ACM Symposium on User Interface Software and Technology (UIST'00)*, 129-137. New York: ACM Press.
- Ward, D.J. & MacKay, D.J.C. (2002) Fast hands-free writing by gaze direction. *Nature* 418(6900), 838.
- Ware, C. & Mikaelian H.H. (1987) An evaluation of an eye tracker as a device for computer input. *Proceedings of the SIGCHI/GI conference on Human factors in computing systems and graphics interface (CHI and GI '87)*, 183-188. New York: ACM Press.
- Wiklund, M.E., Dumas, J.S., & Hoffman, L.R. (1987) Optimizing a portable terminal keyboard for combined one-handed and two-handed use. *Proceedings of the Human Factors Society – 31st Annual Meeting – 1987*, Santa Monica, CA, 585-589. Human Factors Society.
- Wobbrock, J.O. & Myers, B.A. (2006) From letters to words: Efficient stroke-based word completion for trackball text entry. *Proceedings of the 8th International ACM SIGACCESS Conference on Computers and Accessibility (ASSETS '06)*, 2-9. New York: ACM Press.
- Wobbrock, J.O., Myers, B.A., & Kembel, J.A. (2003) EdgeWrite: A stylus-based text entry method designed for high accuracy and stability of motion. *Proceedings of the 16th Annual Symposium on User Interface Software and Technology (UIST'03)*, 61-70. New York: ACM Press.
- Wobbrock, J.O., Rubinstein, J., Sawyer, M.W., & Duchowski, A.T. (2008) Longitudinal evaluation of discrete consecutive gaze gestures for text entry. *Proceedings of the Symposium on Eye Tracking Research & Applications (ETRA'08)*, 11-18. New York: ACM Press.
- Wolfson S. & Case G. (2000) The effects of sound and colour on responses to a computer game. *Interacting with Computers* 13(2), 183-192.

- Yamada, M. & Fukuda T. (1987) Eye word processor (EWP) and peripheral controller for the ALS patient. *IEEE Proceedings Physical Science, Measurement and Instrumentation, Management and Education* 134(4), 328-330.
- Yeo, A.W. & Chiu, P. (2006) Gaze estimation model for eye drawing. *Extended Abstracts on Human Factors in Computing Systems (CHI'06)*, 1559-1564. New York: ACM Press.
- Zhai, S. & Kristensson, P. (2003) Shorthand writing on stylus keyboard. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI'03)*, 97-104. New York: ACM Press.

1. **Timo Partala:** Affective Information in Human-Computer Interaction
2. **Mika Käki:** Enhancing Web Search Result Access with Automatic Categorization
3. **Anne Aula:** Studying User Strategies and Characteristics for Developing Web Search Interfaces
4. **Aulikki Hyrskykari:** Eyes in Attentive Interfaces: Experiences from Creating iDict, a Gaze-Aware Reading Aid
5. **Johanna Höysniemi:** Design and Evaluation of Physically Interactive Games
6. **Jaakko Hakulinen:** Software Tutoring in Speech User Interfaces
7. **Harri Siirtola:** Interactive Visualization of Multidimensional Data
8. **Erno Mäkinen:** Face Analysis Techniques for Human-Computer Interaction
9. **Oleg Špakov:** iComponent - Device-Independent Platform for Analyzing Eye Movement Data and Developing Eye-Based Applications
10. **Yulia Gizatdinova:** Automatic Detection of Face and Facial Features from Images of Neutral and Expressive Faces
11. **Päivi Majaranta:** Text Entry by Eye Gaze