

Summer 2007

Development and analysis of an Internet browsing utilizing the overscan technique for persons with physical disabilities

Patrick Paul Clerkin
Louisiana Tech University

Follow this and additional works at: <https://digitalcommons.latech.edu/dissertations>



Part of the [Biomedical Engineering and Bioengineering Commons](#)

Recommended Citation

Clerkin, Patrick Paul, "" (2007). *Dissertation*. 490.
<https://digitalcommons.latech.edu/dissertations/490>

This Dissertation is brought to you for free and open access by the Graduate School at Louisiana Tech Digital Commons. It has been accepted for inclusion in Doctoral Dissertations by an authorized administrator of Louisiana Tech Digital Commons. For more information, please contact digitalcommons@latech.edu.

DEVELOPMENT AND ANALYSIS OF AN INTERNET BROWSER UTILIZING
THE OVERSCAN TECHNIQUE FOR PERSONS WITH
PHYSICAL DISABILITIES

by

Patrick Paul Clerkin, B.S.

A Dissertation Presented in Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy

COLLEGE OF ENGINEERING AND SCIENCE
LOUISIANA TECH UNIVERSITY

August 2007

UMI Number: 3270941

INFORMATION TO USERS

The quality of this reproduction is dependent upon the quality of the copy submitted. Broken or indistinct print, colored or poor quality illustrations and photographs, print bleed-through, substandard margins, and improper alignment can adversely affect reproduction.

In the unlikely event that the author did not send a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyright material had to be removed, a note will indicate the deletion.

UMI[®]

UMI Microform 3270941

Copyright 2007 by ProQuest Information and Learning Company.

All rights reserved. This microform edition is protected against unauthorized copying under Title 17, United States Code.

ProQuest Information and Learning Company
300 North Zeeb Road
P.O. Box 1346
Ann Arbor, MI 48106-1346

LOUISIANA TECH UNIVERSITY

THE GRADUATE SCHOOL

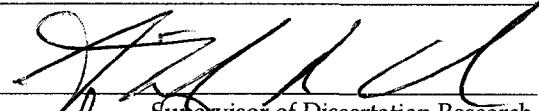
July 6, 2007

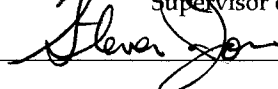
Date

We hereby recommend that the dissertation prepared under our supervision
by Patrick Paul Clerkin

entitled Development and Analysis of an Internet Browser Utilizing the Overscan Technique for
Persons with Physical Disabilities

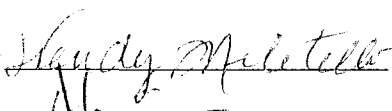
be accepted in partial fulfillment of the requirements for the Degree of
Doctor of Philosophy in Biomedical Engineering



Supervisor of Dissertation Research


Head of Department
Biomedical Engineering
Department

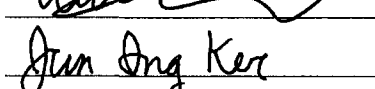
Recommendation concurred in:



Advisory Committee



Advisory Committee



Advisory Committee

Approved:

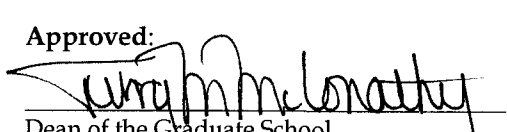


Director of Graduate Studies



Dean of the College

Approved:



Dean of the Graduate School

ABSTRACT

Persons with a severe disability often use scanning as an indirect selection technique for operating augmentative and alternative communication aids and computer access. For information that can be organized in advance, including lists of communication elements such as words and phrases, users often employ rate enhancing scanning methods like the row-column scanning technique. However, row-column scanning requires selection elements to be grouped into defined rows and columns, and therefore does not work well with Internet browsing due to the non-grouped layout of HTML pages.

This work attempts to develop an improved scanning technique for Internet browsing by designing interfaces to compare two contemporary scanning techniques with the overscan scanning technique, also known as the critically damped selection technique. The hypothesis of this investigation is that the overscan technique is a viable technique for persons with a severe physical disability to use to access the Internet. Alphabetic and Internet browsing interfaces were designed to test the error rates, throughput, key press times, reaction times, and activation forces for three different scanning methods: linear, row-column, and overscan. The effectiveness of the interface was determined by testing each interface with individuals without a disability, and a Goals,

Operators, Methods, and Selection Rules (GOMS) model for the three scanning methods tests was developed.

The throughput of the overscan technique was a significant improvement over the linear scan technique for both the alphabetic and Internet interfaces. The individuals testing the interface were able to realize this increased throughput while maintaining error rates which were slightly less than the error rates measured while using the row-column interface. The error rates for the overscan and row-column scanning techniques were greater than the error rates for the linear scanning technique, but the time lost on erroneous selections was much less than the time gained through the use of the overscan and row-column selection techniques.

The overscan technique was shown to be a viable scanning technique for Internet browsing. Use of overscan as a method of indirect selection for Internet browsing could connect individuals to the Internet who are not now linked to this electronic communication medium.

APPROVAL FOR SCHOLARLY DISSEMINATION

The author grants to the Prescott Memorial Library of Louisiana Tech University the right to reproduce, by appropriate methods, upon request, any or all portions of this Dissertation. It is understood that "proper request" consists of the agreement, on the part of the requesting party, that said reproduction is for his personal use and that subsequent reproduction will not occur without written approval of the author of this Dissertation. Further, any portions of the Dissertation used in books, papers, and other works must be appropriately referenced to this Dissertation.

Finally, the author of this Dissertation reserves the right to publish freely, in the literature, at any time, any or all portions of this Dissertation.

Author Patrick Paul Clerkin *PC*

Date July 6, 2007

TABLE OF CONTENTS

ABSTRACT	iii
TABLE OF CONTENTS	vi
LIST OF TABLES.....	viii
LIST OF FIGURES.....	ix
ACKNOWLEDGEMENTS.....	xii
CHAPTER 1 INTRODUCTION	1
CHAPTER 2 LITERATURE REVIEW	5
2.1 Target Population	5
2.1.1 Individuals with Severe Physical Disabilities.....	5
2.1.2 Individuals with CTDUEs	10
2.2 Scanning	12
2.3 Internet Browsing for Persons with a Disability	17
2.4 GOMS Modeling	20
2.4.1 General GOMS Research.....	21
2.4.2 GOMS Modeling in Rehabilitation.....	26
CHAPTER 3 METHODOLOGY.....	33
3.1 Row-Column Scanning Experiments.....	35
3.2 Linear, Row-Column, and Overscan Experiments	40
3.3 Internet Browsing Experiments	44
3.4 GOMS Modeling	52
CHAPTER 4 RESULTS.....	56
4.1 Row-Column Scanning	56
4.2 Linear, Row-Column, and Overscan Alphabetic Interface.....	61
4.2.1 Linear Alphabetic Interface	61
4.2.2 Row-Column Alphabetic Interface.....	66
4.2.3 Overscan Alphabetic Interface.....	70
4.2.4 All Alphabetic Interface Tests	77

4.3 Internet Browser.....	83
4.3.1 Linear Internet Interface.....	83
4.3.2 Overscan Internet Interface	88
4.3.3 All Internet Browser Interface Tests.....	94
CHAPTER 5 ANALYSIS.....	100
5.1 Row-Column Scanning	100
5.2 Linear, Row-Column, and Overscan.....	103
5.3 Internet Browser.....	108
5.4 GOMS Modeling	111
CHAPTER 6 CONCLUSIONS.....	114
REFERENCES.....	116

LIST OF TABLES

Table 2.1 Prevalence of Severe Motor Disabilities in the United States [35]	9
Table 3.1 English Language Frequency of Use	53
Table 5.1 Paired T-Test for First 10 Forces vs. Last 10 Forces	101
Table 5.2 T-Test for Number of Errors Made at JAR and MTF Speeds.....	102
Table 5.3 First Section of Minitab ANOVA Output	103
Table 5.4 Second Section of Minitab ANOVA Output	104
Table 5.5 Third Section of Minitab ANOVA Output	104
Table 5.6 ANOVA Analysis for Linear Alphabetic Scan.....	105
Table 5.7 ANOVA Analysis for Row-Column Alphabetic Scan	106
Table 5.8 ANOVA Analysis for Overscan Alphabetic Scan	106
Table 5.9 ANOVA Analysis for All Scan Tests of Alphabetic Scan.....	107
Table 5.10 ANOVA Analysis for Linear Scan of Internet Browser.....	108
Table 5.11 ANOVA Analysis for Overscan of Internet Browser.....	109
Table 5.12 ANOVA Analysis for All Scan Tests of Internet Browser.....	110

LIST OF FIGURES

Figure 3.1	Screenshot of Row-Column Interface	36
Figure 3.2	Flexiforce Sensor and Excitation Circuit Used to Gather Force Data[18]	37
Figure 3.3	National Instruments SCXI-1000DC and 6036E DAQCard[42]	38
Figure 3.4	Don Johnston Switch Interface Pro 5.0.....	42
Figure 3.5	Tash Membrane Switch.....	42
Figure 3.6	Interlink Electronics Force Sensing Resistor	43
Figure 3.7	First Webpage of Internet Browsing Test	47
Figure 3.8	Second Webpage of Internet Browsing Test	48
Figure 3.9	Third Webpage of Internet Browsing Test	49
Figure 3.10	Fourth Webpage of Internet Browsing Test	49
Figure 3.11	Fifth Webpage of Internet Browsing Test.....	50
Figure 3.12	Completion Webpage of Internet Browsing Test.....	50
Figure 3.13	Missed Link Page Internet Browsing Test.....	51
Figure 4.1	Force measurements for single experimental run.....	57
Figure 4.2	Force Profile for a Single Switch Activation.	58
Figure 4.3	Force of First 10 Activations vs. Last 10 Activations for One Participant	58
Figure 4.4	Force of First 10 Activations vs. Last 10 Activations for Each Test	59
Figure 4.5	Scan Delays deemed "Just About Right" and "Much Too Fast"	60

Figure 4.6	Number of errors committed by each subject at scan delays deemed “Just About Right” (JAR) and “Much Too Fast” (MTF).	60
Figure 4.7	Completion Time for All Linear Tests of Alphabetic Scan	61
Figure 4.8	Mean Force for All Linear Tests of Alphabetic Scan.....	62
Figure 4.9	Key Press Time for All Linear Tests of Alphabetic Scan.....	63
Figure 4.10	Mean Reaction Time for All Linear Tests of Alphabetic Scan.....	64
Figure 4.11	Errors vs. Scan Delay for All Linear Tests of Alphabetic Scan.....	65
Figure 4.12	Completion Time vs. Scan Delay for All Row-Column Tests of Alphabetic Scan	66
Figure 4.13	Force vs. Scan Delay for All Row-Column Tests of Alphabetic Scan.....	67
Figure 4.14	Key press Time vs. Scan Delay for All Row-Column Tests of Alphabetic Scan.....	68
Figure 4.15	Reaction Time vs. Scan Delay for All Row-Column Tests of Alphabetic Scan.....	69
Figure 4.16	Errors vs. Scan Delay for All Row-Column Tests of Alphabetic Scan.....	70
Figure 4.17	Completion Time vs. Scan Delay for All Overscan Tests of Alphabetic Scan.....	72
Figure 4.18	Force vs. Scan Delay for All Overscan Tests of Alphabetic Scan	73
Figure 4.19	Key Press Time vs. Scan Delay for All Overscan Tests of Alphabetic Scan.....	74
Figure 4.20	Reaction Time vs. Scan Delay for Overscan Tests of Alphabetic Scan.....	76
Figure 4.21	Errors vs. Scan Delay for All Overscan Tests of Alphabetic Scan	77
Figure 4.22	Completion Time for All Tests of Alphabetic Scan.....	78

Figure 4.23 Mean Force for All Tests of Alphabetic Scan	79
Figure 4.24 Key Press Time for All Tests of Alphabetic Scan	80
Figure 4.25 Reaction Time for All Tests of Alphabetic Scan	81
Figure 4.26 Errors for All Tests of Alphabetic Scan.....	82
Figure 4.27 Completion Time vs. Scan Delay for All Linear Tests of Internet Scan	83
Figure 4.28 Force vs. Scan Delay for All Linear Tests of Internet Scan	84
Figure 4.29 Key Press Time vs. Scan Delay for All Linear Tests of Internet Scan	85
Figure 4.30 Reaction Time vs. Scan Delay for All Linear Tests of Internet Scan	86
Figure 4.31 Errors vs. Scan Delay for All Linear Tests of Internet Scan	87
Figure 4.32 Completion Time vs. Scan Delay for All Overscan Tests.....	89
Figure 4.33 Force vs. Scan Delay for All Overscan Tests	90
Figure 4.34 Key Press Time vs. Scan Delay for All Overscan Tests	91
Figure 4.35 Reaction Time vs. Scan Delay for All Overscan Tests	93
Figure 4.36 Errors vs. Scan Delay for All Overscan Tests.....	94
Figure 4.37 Mean Completion Time for All Tests of Internet Scan	95
Figure 4.38 Mean Force for All Tests of Internet Scan	96
Figure 4.39 Key Press Time for All Tests of Internet Scan.....	97
Figure 4.40 Reaction Time for All Tests of Internet Scan.....	98
Figure 4.41 Errors for All Tests of Internet Scan	99

ACKNOWLEDGEMENTS

I would like to thank Dr. Stan Cronk for his incredible support throughout my entire journey as a graduate student. Dr. Cronk has shown immense encouragement, leadership, and patience during my graduate years, and I am humbled to have him as a mentor. I will always consider him not only a mentor, but also a lifelong friend. I would also like to thank Louisiana Tech University and the staff at the Center for Biomedical Engineering and Rehabilitation Science for providing me with the unique opportunities to perform research and practice clinical services at the same time.

Furthermore, I would like to thank my exceedingly generous parents, John and Beth Clerkin. I could not ask for more understanding and charitable parents. Their financial and personal support has enabled me to realize my goals. I can only hope that if I become a parent someday, I will be able to help my children as they have helped me.

CHAPTER 1

INTRODUCTION

Individuals with severe disabilities often use Assistive or Adaptive Technology to augment or replace many tasks that individuals without disabilities perform without any assistance. In 1998, the United States Congress passed the Assistive Technology Act of 1998. This law defined an assistive technology device as *any item, piece of equipment, or product system, whether acquired commercially, modified, or customized, that is used to increase, maintain, or improve functional capabilities of individuals with disabilities*. This broad definition encompasses all types of assistive technology including Activities for Daily Living (ADL) devices, Augmentative and Alternative Communication (AAC) devices, computer software, computer hardware, environmental control, orthotics, prosthetics, seating, and wheelchairs.

Barriers and limitations which were previously thought to be insurmountable for persons with disabilities are continually overcome as assistive technology continues to evolve, enabling more and more of the population with disabilities to achieve greater functionality and independence. Assistive technology devices currently allow individuals with disabilities to participate in the home, classroom, workplace, and community by overcoming

functional limitations that previously would have limited individuals with disabilities to assisted living institutions or a quiescent life stuck at home. For example, in the past, individuals with visual disabilities were excluded from scholastic endeavors where math played a key role, such as engineering and science. The few individuals who were did enter into such programs were often excused from participating in any exercises which required graphical or visual content, and therefore did not get the same education as individuals without a disability. However, blind individuals can now be included in the classroom thanks to researchers who have augmented visual subject matter with audio and haptic material. In the classroom, assistive technology such as screen readers, electro-mechanical Braille displays, calculators with audio output, and optical character recognition programs have all been used to include persons with a visual disability. Without advances in assistive technology, these individuals would not be free to pursue all of the same math and science academic disciplines that individuals without a disability pursue.

Unfortunately the development of assistive technology often lags far behind new and emerging technologies. One such technology is the Internet, also known as the World Wide Web. The 1990s saw the Internet evolve from an arcane tool used primarily by DARPA, the Department of Defense, and select research communities, to a ubiquitous part of American society. As the number of people using the Internet grew, more and more uses for this technology developed. As this technology and its uses developed, the way society accessed

information and communication changed forever. For example, people started communicating through electronic mail, checking the weather, getting driving directions, paying bills, and even ordering pizza through the Internet [6].

Regrettably, people with a disability were unable to make use of this new technology because the contemporary computer interfaces include many barriers to access. Modern computer interfaces require the adroit use of not only a keyboard, with a finite number of keys, but also a pointing device such as a mouse that can be positioned in an almost infinite number of ways. The fact that persons with a disability did not have equal access to the Internet was especially egregious because the Internet offers so much potential to increase the independence of persons with disabilities. The Internet also opens employment opportunities to individuals with a disability who are not easily able to leave their homes [20].

This work is an examination of a little used AAC technique, overscan, which was developed in the 1980s but failed to catch on due to a variety of reasons discussed in the Literature Review section of this dissertation [3]. This technique was not chosen for a purely academic exercise to study an older technique, but rather this technique was chosen because overscan has great potential to work as an interface for newer technology that was not in common use when this technique was being used. This project attempts to combine this AAC technique and the Internet in a manner that will enable users with physical disabilities to connect with the rest of society via the World Wide Web. While

other researchers and developers have attempted to give this unique population a viable method of accessing the Internet, this new method will give these users a web browsing experience which is cost-effective, efficient, and visually similar to the browsing experience of individuals without a disability. In order to develop an Internet browsing technique that is efficient enough to be considered viable, the overscan technique will be investigated as a possible method of scanning and selecting the HTML links on a web page.

CHAPTER 2

LITERATURE REVIEW

2.1 Target Population

2.1.1 Individuals with Severe Physical Disabilities

In 1998, individuals with a disability were about one quarter as likely to use the Internet as individuals with no disability [32]. The 2000 Census data shows that 6.2 percent of the population aged 16 to 64 years lives with a physical disability [16]. Fortunately, many individuals with a disability now have access to computers through the use of adaptive interfaces such as screen readers, speech recognition, and special keyboards [4]. However, individuals with a severe motor disability still do not have an effective, low-cost interface for accessing the Internet. Previous attempts to provide fully accessible access to the Internet for persons with disabilities have not been effective for three reasons.

- The assistive technology has been too expensive for many individuals, many of whom are impoverished.
- The assistive technology does not meet the specific needs of individuals with severe disabilities, and instead attempts to provide access to individuals with a wide range of disabilities.

- The assistive technology is so tedious and cumbersome that access becomes very slow and frustrating, and the technology is not adopted by the individuals it was designed to help.

An example of the first reason is any technology which is very expensive and requires significant technical support to maintain, such as a Brain-Computer Interface (BCI). Devices have been developed and studied for more than 30 years which utilize a user's brain waves to communicate with a computer. Researchers have been testing different ways to get information from the brain, different brain communications to measure, and the speed at which the information from the brain can be used to communicate with a machine. While this technology is very promising and continues to evolve, current and previous interfaces have not been cost effective. These BCI devices require very expensive hardware such as customized electrodes and advanced signal processing systems. Furthermore, these devices always require significant professional training and dedicated technical support. Most individuals with a severe disability do not possess the resources to acquire these devices without outside financial help, and the high cost of the interfaces means that government agencies who provide assistive technology to persons with disabilities cannot afford to purchase these interfaces[41][50][52].

The second reason that previous attempts at providing Internet access have not been effective for persons with severe physical disabilities is that some of these attempts have been targeted at individuals with a wide range of

disabilities, and these technologies targeted at large target populations do not meet the specific needs of individuals with severe physical disabilities. Speech recognition technology has enabled millions of individuals to interact with a computer system to perform a variety of tasks using only vocal input. Recent commercial versions of speech recognition software have become so accurate that individuals no longer need to spend time training the software, and the individual can begin interacting with the computer right away. However, speech recognition technology does not adequately meet the needs of many users with severe physical difficulties because these individuals often do not possess the vocal quality necessary to efficiently interact with a speech recognition interface [47].

Finally, the third reason that previous attempts at providing Internet access have not been effective for persons with severe physical disabilities is that current technologies directed towards individuals with severe disabilities are so slow that this technology has not been adopted and embraced by the individuals who use the technology, or by the clinicians who prescribe and set up the technology. Current interfaces for persons with severe physical disabilities often fall into this category. Individuals with severe physical disabilities are often not capable of using a pointing device like a mouse or a trackball to directly select the desired hyperlink on a web page. Some interfaces attempt to circumvent this issue by using an indirect selection technique such as scanning to select the desired link. Many current scanning interfaces which scan through Internet links

do so at such a grueling pace that the throughput, or time to select a link, is intolerably long. If a user wants to select a link at the bottom of the page, he or she might need to scan through over 100 links before arriving at the desired selection.

While these three reasons have prevented many individuals with severe disabilities from effectively accessing the Internet, the need for access by these individuals continues. An improved Internet scanning interface will support individuals with disabilities, maximizing integration into society by connecting them with an essential aspect of the social fabric of American life. Recent research has shown that the Internet builds social networks by giving individuals a medium to sustain dynamic communication with large social networks of people who do not necessarily live close to one another [4]. This method of communication not only transforms how individuals without a disability interact with society, but the Internet also offers an even greater potential to vastly increase the social networks of individuals with disabilities. Individuals with severe physical disabilities who have access to the Internet will be able to circumvent traditional barriers by using the Internet as a communication medium. This communication medium will allow individuals who are non-ambulatory to communicate without relying on others to transport them or facilitate communication in other ways. The ability to communicate through the Internet means that individuals with disabilities will be able to communicate

from the comfort of home. This social self-sufficiency will vastly increase the quality of life for individuals with severe disabilities.

Access to the Internet will also open up employment opportunities that individuals with disabilities would not otherwise have. There are also many resources on the Internet to help individuals with disabilities find employment. Jobs that require an individual to access the Internet are virtually inaccessible to persons in the target population. By creating a more efficient Internet scanning interface, individuals with severe disabilities will gain greater access to jobs thereby facilitating employment and economic self-sufficiency. Table 2.1 shows the number of individuals who experience the most common forms of motor disability in the United States:

Table 2.1 Prevalence of Severe Motor Disabilities in the United States [35]

Disability	Number of Individuals
Multiple sclerosis	226,000
Cerebral Palsy	211,000
Amyotrophic lateral sclerosis	105,000
Partial Paralysis of Upper Extremity	80,000
Paralysis of Upper Extremity	47,000
Quadriplegia	44,000

While researchers disagree on the exact numbers of individuals affected by these serious motor disabilities, it is certain that a sizeable percentage of such individuals would benefit from a scanning interface that would provide access to the Internet. When the prevalence of these severe disabilities is coupled with the fact that individuals with severe disabilities are less likely to access the Internet, there is clearly a large need for an Internet interface designed specifically for individuals with a severe physical disability [17][21][41].

2.1.2 Individuals with CTDUEs

Individuals with a severe physical disability are not the only persons that would benefit from a scanning Internet interface. People with repetitive stress injuries, or more specifically Cumulative Trauma Disorders of the Upper Extremities (CTDUEs), must limit the time they use a mouse and keyboard. It is estimated that two to three percent of the US population suffers from carpal tunnel syndrome [35], the most limiting of all CTDUEs. An even higher percentage of the population suffers from this and other types of upper extremity musculoskeletal disorder [7] [36]. In 2005, the Bureau of Labor and Statistics reported that carpal tunnel syndrome caused workers to miss more days of work than any other major disabling injury or illness. The bureau also reported that carpal tunnel syndrome and tendonitis combined to cause 2.1 percent of the nonfatal occupational injuries reported in 2004 [51]. The estimates from the report from the Bureau of Labor and Statistics are likely conservative because university campuses have high incidences of CTDUES, and the student

population is not included in these statistics because not all students are counted as part of the workforce[45][8].

Research has shown that as time spent using input devices such as a keyboard and mouse increases, so does the incidence of CTDUEs [5][10][26]. Physicians disagree on the best treatment for CTDUEs, but most agree that rest is an essential aspect of any treatment regimen [54] [55]. Since much computer use involves accessing information on the Internet, the use of a scanning web browsing interface should eliminate a large amount of a user's time spent typing or operating a mouse. It is also important to note that mouse use shows the highest correlation of CTDUE development, and most Internet browsing involves a significant amount of mouse usage [25][34].

Other methods are commercially available for persons with CTDUEs to access the Internet. One example is the use of Speech Recognition software. By using this software, individuals can navigate the Internet through vocal commands; however, there are two problems with this approach. First, many current speech recognition programs added web navigation as an afterthought, and many programs require the user to emulate the mouse with vocal commands. Some programs, like Nuance Communications Dragon NaturallySpeaking, do let the user directly select links with vocal commands by numbering each link on the screen. However, many links are not handled correctly, and users must revert to the vocal mouse emulation method [31] [40] [47]. The second problem with this technology is that individuals with CTDUEs

are often prone to developing other repetitive stress disorders. Research has shown that users with CTDUEs who switch to speech recognition may develop repetitive stress injuries of the vocal cords, compounding the effects of their disability [30].

The intensity level and duration of this disability can often be lessened from adequate rest of the upper extremities. Unfortunately, the omnipresent nature of computers and the need to access the Internet in contemporary society does not allow individuals to simply stop using the World Wide Web because they are suffering from CTDUEs. Individuals who rely on the Internet for work cannot simply stop using the World Wide Web for the many months or years of required rest without significant financial hardship. This leads to the need for an alternate Internet interface which allows individuals to continue accessing the Internet without using the same repetitive motions which caused the CTDUEs. This alternate interface would need to allow individuals with CTDUEs to temporarily limit the amount of time using a traditional pointing interface, or even stop using this interface altogether.

2.2 Scanning

Scanning is an indirect selection technique with widespread use in the rehabilitation field [4] [1]. The scanning technique involves scanning through elements of a selection set one element or group at a time in a predefined pattern. The elements of the selection set are presented by presenting or highlighting one

element or group of elements for a set amount of time and then moving to the next element or group of elements in the selection set. When the desired element is presented or highlighted, the user selects that item by pressing a switch or making some other signaling motion. Originally, this technique was implemented by a trained communication partner, or facilitator, pointing to each element until the user gave a signal that the facilitator was pointing at the correct element. Later, electronic devices took the place of the facilitator and highlighted elements of the scanning matrix presented on a screen. This technique is well suited for individuals with severe disabilities because the electronic device can be controlled with only one signaling event. This means that an individual who is only able to blink his or her eyes could still use the electronic device by employing the scanning technique. Other signaling methods include sip/puff switches, membrane switches, rocker switches, tongue switches, and infrared proximity switches. This wide array of possible switch interfaces allows users with many different disabilities to access electronic devices.

Currently, scanning is used primarily by human-machine interfaces for augmentative and alternative access devices such as the DynaVox Technologies DV4 or the Prentke Romich Company Vanguard. These devices are stand alone machines which enable the user to communicate through the use of one or more switches. More recently, scanning has been used to control computers through interfaces which emulate keyboards and pointing devices. This newer

implementation opens up many new possibilities to individuals with physical disabilities because computers are such an integral part of modern society.

The scanning technique has been well studied by researchers in the fields of rehabilitation, augmentative and alternative communication, and biomedical engineering. Because scanning is a very slow selection technique, many researchers have looked at various methods of increasing the communication rates of individuals using scanning for computer access or communication. While communication rates for individuals without a disability vary from 100 to 200 words per minute (WPM) for spoken language, and 35 to 40 WPM for typing on a keyboard, individuals using scanning interfaces often communicate at rates of 5 WPM or less [22]. Since the communication rate is very low using the scanning technique, it is essential that the rate is maximized to reduce frustration and mental fatigue. Optimization of the scanning interface becomes even more essential for individuals who use scanning as their sole method of communicating with the world.

Changing the scanning technique is one way of increasing the throughput. One researcher found that row-column scanning had twice the throughput of linear scanning [53]. Another method of increasing the throughput is to change the scanning matrix. Researchers have looked at changing the shape, size, number of dimensions, and layout of the scanning matrix in order to increase the efficiency and throughput for individuals using the interface[53][1][38]. By changing the scanning matrix, researchers have attempted to give users the most

efficient layout for optimal and relatively quick communication. For example, scanning matrices have been designed so the letters are arranged by frequency of use [53] [38]. However, although this layout provides users with a quicker method of scanning through the alphabet, some users prefer the traditional alphabetic or QWERTY layouts. Because of the unique nature of individuals with disabilities, user preference is often paramount in the decision of which layout to use [19].

Optimizing the scan delay, or time spent between elements of the matrix, is another method of increasing the user's throughput. Researchers have attempted to adapt the scan delay automatically, using computer algorithms. Research groups have looked at various aspects of the scanning experience to determine which aspects of the scan could be recorded by the scanning interface and used to automatically adjust the scanning parameters. In 1987, Cronk and Schubert attempted to use the structure of Expert System Technology, from the domain of artificial intelligence, to develop an interface which automatically adapted to the user's ability to operate the scan [11]. This interface attempted to use input parameters such as error rates and the reaction time to compute an efficient scan delay for the current user. The researchers developed this interface algorithm by observing the changes made by clinicians working with persons with severe disabilities, and attempted to mimic the strategies employed by the clinicians used for these changes. This technique of automatically adjusting the scan delay based on error rates has the capacity to find the most efficient scan

delay. However, researchers found that users often do not like having the scan delay adjusted automatically, and would rather adjust the delay to a speed which is the most comfortable. In the area of AAC, user preference often supersedes communication rate, and it is essential that the users feel comfortable with the speed of the scan delay instead of simply choosing the scan delay which will give the best throughput.

Cronk continued his work looking at aspects of the scanning experience by looking at user satisfaction at various scan delays [12] [13] [14] [15]. This work attempted to determine which scan delay was the most preferable to individual users, and determine the relationship of this scan delay to error rates, key press force, and reaction times.

Recently, researchers have resumed looking at interfaces which make decisions about the scan delay period without input from the user or a clinician [49]. Simpson et al. looked at an adaptive scanning system and evaluated the Input Device Agent (IDA). This IDA system makes decisions about the scan delay of a scanning device for individuals using a scanning interface. These researchers found that this system produced a communication rate that was as good as or better than the communication rate achieved when individuals selected their own scan delay parameters.

The overscan technique is a scanning technique where the scan delay is set at a speed which is faster than the user could accurately select the desired target element. In order to select the desired element, the user must let the scanning

interface pass the target and hit the switch to reverse the scan. Once the scan is reversed, the scan delay is set to a slower speed which allows the user to accurately select the target element. The purpose of this quick scan delay is to increase the throughput for the scanning interface, but this interface did not become popular when it was first introduced because the row-column scanning technique was the preferred method for rate enhancement. However, Internet web pages are filled with links that do not work well with row-column scanning because links are not grouped into well-defined rows and columns. Conversely, the overscan technique does not require scanning elements to be grouped, and this work seeks to determine if the overscan technique is a viable method of navigating the Internet.

2.3 Internet Browsing for Persons with a Disability

While many of these individuals do have computer access through scanning interfaces, these interfaces are very awkward for accessing the Internet because the interfaces were not designed specifically for Internet browsing. Instead, these interfaces were designed for general computer access. Most of these scanning interfaces require the user to navigate through the links on a web page by emulating a computer keyboard and mouse with static menus which are not context-sensitive, requiring the user to scan through many unnecessary elements in order to browse the Internet.

This method of surfing the Internet is much too slow and tedious, and the rehabilitation community has identified the need for a better method of Internet access for individuals with severe physical disabilities [41]. Internet access needs to become more efficient for persons with a disability so these individuals can have a more useful and enjoyable Internet experience. Researchers have taken different approaches to developing an Internet browser for persons with a physical disability.

One group developed an Internet browser which modified the minimum size for clickable images so that individuals with physical disabilities could more easily position the mouse over the image and select the image [21]. This improved accessibility for some users, but it still required users to use a keyboard and mouse. While this improved accessibility is important for users that are able to use a keyboard and mouse, individuals with severe physical disabilities or individuals with CTDUES must rely on other methods for accessing the Internet. Other researchers have attempted to develop an Internet browser utilizing the scanning technique, but instead of developing a scanning interface which handles all aspects of computer access these new interfaces were designed specifically for Internet accessibility.

In Spain, researchers developed a scanning interface specifically designed for web browsing [17]. This design uses an interface tool, Switch Access to Windows (SAWS), which allows the user to create custom scanning interfaces. This improved interface takes the typical mouse and keyboard emulating

technique a step further by eliminating any unnecessary elements necessary to surf the Internet, and also putting in elements specifically needed for web access. These additional elements include the ability to move from link to link, and the ability to print a web page by selecting a single element in the scanning matrix.

This interface was designed with SAWS, therefore the interface simply emulates the keyboard and mouse functioning by sending scripts and keystrokes to the machine. This interface uses Internet Explorer as the browser, but does not integrate directly with the browser. If future versions of Internet Explorer change the keyboard shortcuts for tasks, commands selected using this interface would stop working or have unexpected consequences. The interface menu structures are not context-sensitive. Finally, the interface does not automatically scan through links; it only provides a "go to next link" function which must be repeatedly selected to navigate to the desired link.

2.4 GOMS Modeling

The science of modeling Human Computer Interfaces (HCI) has been around for over 20 years; however, very little research time has been spent modeling the interaction between a user and various alternate interfaces. Individuals with a severe disability use these interfaces as a way to control devices for communication, a computer, or environmental control units. This lack of research means that developers and clinicians alike do not have a way of quantifying how users interact with a computer or other electronic devices. Without the ability to measure and quantify such interaction, progress is greatly slowed in the field of AAC as well as other rehabilitation fields involving computer or electronic access.

HCI modeling is a description of the interaction that takes place between a human and a computer through a given interface. This model is then used to analyze and improve the way humans interact with machines through interfaces. The GOMS model in particular has been around since 1983 when Card, Moran, and Newell proposed the technique as a way of evaluating human performance [9]. They introduced a psychological model and called it the Model Human Processor which was defined as a set of memories and processors together with a set of principles. This model was divided into three interrelated subsystems: the perceptual system, the motor system, and the cognitive system. The perceptual system is the part of the model that involves observation by the body's visual system in response to human-computer interfaces. The motor system is the

voluntary muscle actions, or effectors. Finally, the cognitive system is the connection between the two other systems, and it also involves some processing of memory.

2.4.1 General GOMS Research

In 1996, Bonnie E. John wrote an article which provides a good overview of GOMS modeling because the article discusses what GOMS is, what it does, where it applies, and its value [29]. To begin, the article describes GOMS by breaking it into Goals, Operators, Methods, and Selection Rules. This article also discusses the different variations of GOMS from the original GOMS all the way to newer techniques like CPM-GOMS. CPM-GOMS uses cognitive, perceptual, and motor operators in a critical path method schedule chart (PERT chart). This new form of GOMS gets over a problem with the original GOMS by allowing for activities to be performed in parallel.

The article describes how GOMS analysis produces quantitative and qualitative calculations to predict how users will interact with a system. The quantitative predictions estimate elements like execution times, wait times, and learning times. Qualitative predictions include ease of use and likelihood of errors. This article ends by giving an example of the value of the GOMS model where researchers improved the performance time of a routine task for a map digitizing system. This improvement allowed them to predict a cost savings of \$730,000 for the company [29].

John collaborated with David E. Kieras to write another article on GOMS, which is a lengthy article that examines four variants of GOMS. These four variations are: the original GOMS, Keystroke-Level Model (KLM), CPM-GOMS, and NGOMSL (Natural GOMS Language). This article uses one task, editing a marked-up manuscript, to show how each GOMS technique analyzes this task. While explaining how each technique would apply to the task, the authors show how KLM is very simple, followed by the original GOMS, which is slightly more complicated. NGOMSL follows with an elaborate sequential architecture, and finally CPM-GOMS adds multiple parallel processor architecture [27].

This article explains the advantages and disadvantages of each version. For example, the KLM model allows the researcher to analyze an interface in a short amount of time with relatively little effort. On the other hand, CPM-GOMS is an extensive technique that requires in depth analysis and more time, but which produces more accurate results because this technique accounts for parallel processing which is left out of the KLM model. NGOMSL gives the most complete analysis of a system because this technique not only predicts execution times, but also predicts learning times which are very useful when discussing acceptance of one particular system over another.

John and Kieras also wrote a manual which details how an analyst can decide if GOMS is appropriate, and if so, which version should be used [28]. The article discusses the broad concept of engineering models for computer system

design, how GOMS is an example of an engineering model, how GOMS can be applied, and case studies of how GOMS has been applied in the past.

The authors describe how GOMS is meant to predict human performance using a system before a prototype of the system even exists. In this way designers can forgo many expenses involved in the cyclical process of creating a prototype, testing the prototype, and improving the design. Using the GOMS model means that designers can test their system while the design is still just an idea.

The main purpose of this manual was to help designers decide which GOMS technique is appropriate based on what the designers want to predict. The authors provided an excellent graph for determining which variant is the best for a particular situation. The authors provided a clear and concise tool for determining which GOMS to use based on the six design information qualifiers. The six design information qualifiers are: coverage, consistency, operator sequence, execution time, procedure learning time, and error recovery. The tool gives the appropriate GOMS model for each design information qualifier depending on whether or not the task type is sequential or parallel. For example, when modeling execution time for a sequential task, the tool states that researchers should use KLM, CMN-GOMS, or NGOMSL. However, if the task is sequential, CPM-GOMS should be used instead. This tool is an excellent method of determining which variant of GOMS is appropriate for the desired application.

The tool also clearly shows which tasks do not have an appropriate GOMS technique, and therefore require a different modeling technique. For example, if a researcher or designer wanted to model procedure learning time of a parallel task, there is not an appropriate variant of GOMS which would produce an accurate model. In this case, the researcher or designer would need to use a different modeling technique because if a GOMS model was developed the results would not be accurate. This aspect of the tool is very valuable because it is essential to determine whether or not the task being modeled has an appropriate GOMS technique.

The manual also discusses what information is not provided by the various GOMS techniques. The authors state that standard human factors issues such as readability of letters and words on the screen are not addressed by GOMS, but must be dealt with by designers in order to gain a full understanding of the effect. The article concludes by giving many examples of the four variants of GOMS being applied to systems and interfaces. These examples include case studies of actual systems that were designed and optimized using the GOMS modeling technique. The case studies show real world applications of the GOMS modeling technique.

In 1996 Atwood et al. wrote an article which is an in depth analysis of a GOMS application. This article looked at a project where GOMS was used to evaluate the toll and assistance operator (TAO) workstations for a telephone company [2]. The company NYNEX decided to replace the TAO workstations

with new units that were supposed to decrease call time by 4.1 seconds. However, after the new workstations were installed, the call time actually increased by 0.8 seconds. Researchers used CPM-GOMS to analyze the differences in the workstations to see if the predicted difference in execution times matched the empirical data. The researchers found that the CPM-GOMS model predicted an increase in call time of 0.6 seconds which is very close to the empirical data.

The GOMS model explained why the newer system was slower even though this was counter intuitive. The new workstation used more advanced technology to communicate with the switchboard at a higher speed, a new display had a graphical user interface instead of an alphanumeric interface, and a new keyboard placed the most frequent keys closer together. The CPM-GOMS model showed that critical paths in the system accounted for the slowdown in overall call length. These critical paths never included eye movements, and rarely included movement to the correct keys. Because the critical paths did not include parallel processes, a significant slowdown was observed.

In 1994 Shum et al. wrote about the process of transferring HCI modeling from the academic to the practical world. This work describes research into the transfer of analytic HCI approaches to designers [48]. This is very important research, because unless information can be transferred from researchers to designers, there is little hope of any practical significance coming from HCI research. The writers discuss two modeler-designer workshops and a modeling

evaluation tool. The workshops were used to gather information about what questions designers had of modelers, and how modelers see themselves fitting into the larger picture. The modeling tool study investigated what kinds of knowledge of the underlying modeling approach are required in order to use the expert system design tool.

2.4.2 GOMS Modeling in Rehabilitation

The first four articles of this section compose an interesting academic conversation on the modeling of alternative and augmentative communication devices. In 1990, Heidi Horstmann and Simon Levine published an article in the journal, *Augmentative and Alternative Communication*, which started the academic conversation [22]. In this article, the authors apply the GOMS model to three different interfaces: row-column letter scanning, row-column letter scanning with word prediction after the first two letter selections only, and row-column letter scanning with word prediction after each letter selection. By applying the GOMS model to the three interfaces, the authors were able to describe what had been seen by many practitioners but was counter-intuitive: word prediction combined with scanning actually slowed many users down compared to simple scanning. The authors were also able to quantify learning times using the model, and they found that the learning time for simple scanning was only about 80% of the time required to learn the other two methods.

The authors showed that the simple row-column had only seven statements in the formulation of the GOMS model whereas the other two

methods had considerably more. Increasing the number of statements to be executed increases execution time. The word prediction systems also had additional mental operators like visual search time and word matching that are not present in simple row-column scanning. Thirdly, the word prediction success of 70 to 75% in the word prediction systems is not enough to counteract the overhead of the additional mental overhead [22]. Finally, many words entered into the system were simply too short to have a keystroke savings from the word prediction.

In 1992, Newell et al, the creators of one of the word prediction scanning systems wrote a response to Horstmann and Levine's article. In this article, the authors do not agree with the assessment by Horstmann and Levine that the model is a correct representation of how users with a disability interact with the system [44]. The authors point out that Horstmann and Levine used able-bodied subjects to test their systems when individuals with a disability have very different ability levels which would be difficult or impossible to correctly model. They also point out that the GOMS model is based on simple linear addition of times when there is "little evidence that the tasks necessary for operating a scanned matrix are performed sequentially" [44].

The authors point out that users of different ability levels will have different levels of success with a predictive system. For example, users with trouble spelling could benefit greatly from using a predictive system. The authors also discuss fatigue as it relates to character entry rate. For individuals

with a disability, fatigue can play an important role in error rates and character entry rates, especially in disabilities like cerebral palsy. The authors point out that the users in their tests actually had a 50% character entry savings versus a standard scanning system that will greatly reduce the level of user fatigue. Finally, they comment that for very slow conventional keyboard operators (less than 5 words per minute), the increase in speed is approximately equal to the keystroke savings, which is 50% in this case.

In response to the criticism of their initial article, Horstmann and Levine wrote another article [23]. In this article the authors point out that their GOMS model is an initial step into a lengthy modeling process. They feel that by modeling the Augmentative and Alternative Communication (AAC) systems, they have gained much in the way of qualitative analysis of how users interact with the various systems. They also feel that the modeling is much better than the alternative, which are simple empirical studies and intuitive notions. Finally, the authors point out that their previous article was not an attempt to discredit word prediction systems, but rather to “provide a balanced presentation that permits the interested and knowledgeable readership of AAC to draw their own independent conclusions in regard to the significance and implications of our work.”

The last article in this academic argument was written by Newell et al. [43]. In this article, the authors agree with many aspects of Horstmann and Levine’s defense of the initial article, but state that a wide range of communicate

rate increases are to be expected from users with identical keystroke savings. They state that this is to be expected because slower users will gain more from rate prediction than faster users. They point out that this was the reason to implement word prediction in scanning systems, because many users with a disability have very slow scanning speeds and can therefore greatly increase speeds using word prediction. They agree that “accurate prediction can be made without going to the lengths of making a precisely accurate behavioral mode” [43]. However, they feel that in using incorrect models, analysts can produce wildly inaccurate results. They reiterate that they feel it is necessary to use a great deal of caution before using the results of models to analyze AAC systems.

The next article reviewed was also written by Horstmann and Levine, and this work developed a model of text entry for word prediction. This article is a lengthy discussion of why text generation rates for spinal cord injured subjects is decreased when word prediction is enabled [24]. For this article, the authors tested six subjects with high-level spinal cord injuries and eight individuals without a disability. Both groups used a system of a mouthstick keyboard with and without word prediction. The text generation rates for able-bodied individuals increased slightly with word prediction enabled, and the article seeks to explain the difference between the two test groups. The main focus of the article is on the cognitive cost of adding word prediction to the interface, and how this affects able-bodied and individuals with a disability differently.

While both test groups had an increased cognitive load from the addition of word prediction, this increased load had a much greater impact on the communication rate of individuals with a disability. The researchers developed a flow chart modeling the use of a word prediction system. The researchers felt that the discrepancy in the communication rates was accounted for in the steps of searching the word list. Individuals with a disability have a much longer search time because manipulating the word list requires head movement which is much more difficult for an individual with a disability. The researchers also hypothesized that the cognitive load increased more for individuals with a disability because those individuals were used to typing without using word prediction and therefore had become accustomed to very little cognitive load. The addition of word prediction created a much higher cognitive load for these individuals that increased their communication rates. The users without a disability did not experience as great of a cognitive load because they were not used to the mouthstick keyboard as either a word prediction system or a traditional system. No training was provided to them before the testing commenced. This meant that the individuals without a disability did not have to suffer from the same type of qualitative shift in information processing [24].

Keates et al. wrote an article discussing the differences between users with different disabilities [33]. In the article, the authors point out that there is a great deal of difference in the way users with different disabilities interact with computers. The authors argue that the previous modeling techniques involving

testing with able-bodied individuals and then inferring that data onto how individuals with a disability will use various systems is flawed. The paper not only points out how other methods are flawed, but also suggests steps to improve previous methods by understanding how motion-impaired users interact with computers.

The authors start by acknowledging the difficulty in testing and designing models of systems using individuals with a disability. They note that individuals with a disability are harder to find and that user trials can be much more expensive than when using individuals without a disability. Also much more time must be spent collecting data because the trials take longer, and the design team usually has to go to the user instead of getting users to come to them. This difficulty in finding an adequate number of individuals with a disability means that it is often necessary to use individuals without a disability when testing.

The authors attempt to test individual components from the GOMS models of individuals with and without a disability. They tested keystroke times, pointing times, homing times, drawing times, and mental operation times for individuals with and without a disability. The authors found that individual components of the GOMS models were comparable for both able-bodied and individuals with a disability, but that the largest observed difference was in motor function times. Individuals with motion impairments were found to be 50% slower than individuals without a disability. The data gathered by these authors is instrumental in applying GOMS models developed using test subjects

without a disability to individuals with a disability who will ultimately be the ones using the AAC systems.

CHAPTER 3

METHODOLOGY

The goal of this work is to examine the AAC technique of overscan, and determine if this is a suitable method for Internet access for persons with a physical disability. Three sets of experiments were designed to compare the overscan technique with row-column scanning and linear scanning, and the use of scanning to browse the Internet. The following parameters were examined in each set of tests:

1. *Reaction Times*. The time that it takes each individual to activate the switch after the scan highlighted the desired selection was measured in each test. The VB.NET interface measured and stored reaction times automatically along with other data in an Excel data file. Researchers have previously found that users find a scanning rate to be “comfortable” when they activate the switch about 60% of the total scan delay, or time that the scan spends highlighting each item[12][37]. To date, no one has studied reaction time data with the overscan technique to determine the optimal rates of “fast-forwarding” through the links, and then scanning backward. It is important to estimate the optimal scanning rates based on user skill levels with interacting with

a scan, because scanning at rates too fast for the user frustrate the user through excessive errors, while scanning too slowly creates frustration because of the needless delays imposed by the scan.

2. *Error rates.* The differences between the target selections and the actual selections made by each user were recorded for each individual switch activation. Recording the number of errors made for each scanning technique is essential because this error rate gives insight into the efficiency and ease of use of the different techniques. Additionally, the error rates help in gauging which scan delay speeds are too rapid for individuals to efficiently operate the scanning interface.
3. *Forces used to operate the activation switch.* The forces used to operate a control switch for the scan were measured using force sensors connected to an excitation circuit which energized the sensor as well as amplified the output. This output was sent to a National Instruments SCXI signal conditioning unit. This system was controlled through a virtual instrument created in National Instruments LabVIEW 7 data acquisition software, which collected the force data, and wrote the data to log files for later analysis using MATLAB 7.01, Release 14. Force data was recorded to determine what effect various parameters of the scanning interface had on the amount of force users used to activate the switch.

4. *User satisfaction.* After each test, users were asked a series of questions aimed at understanding their satisfaction with the interfaces and operating parameters of the interface. These questions were asked to determine how each individual user perceived the speed of each scanning interface. This set of questions was asked to gauge the users comfort level with the speed and type of scan for each test.

Additionally, at the end of each set of tests, the user was asked which scanning method he or she would prefer to use. This final question is essential because user preference is very important with AAC devices. If the user does not feel that the scanning interface given to him or her is the best method, he or she is not likely to use this interface in the future.

3.1 Row-Column Scanning Experiments

In order to determine if switch press forces, switch press times, error rates, and reaction times are related to changing scan delays, the first set of experiments were designed to examine these parameters for the most commonly used scan technique, row-column scanning. The first step in the design of this experiment was to develop the scanning interface.

A scanning interface previously developed at the University of Tennessee was modified to work with Visual Basic .NET 2003[12]. This interface employed the traditional square matrix layout, and all characters were presented in

alphabetical order. This interface used a single switch control, which was the left mouse button of the touchpad of the Dell Latitude C400 laptop computer, used to run the program.

This interface recorded the selected element, reaction time, a timestamp, and an error code for each switch activation made by the user operating the program. The error code was used to determine whether or not an error was made, and if so what type of error was made. This data was then written to an Excel spreadsheet so that the data could be analyzed offline. Figure 3.1 is a screenshot of the interface running in Visual Basic .NET.

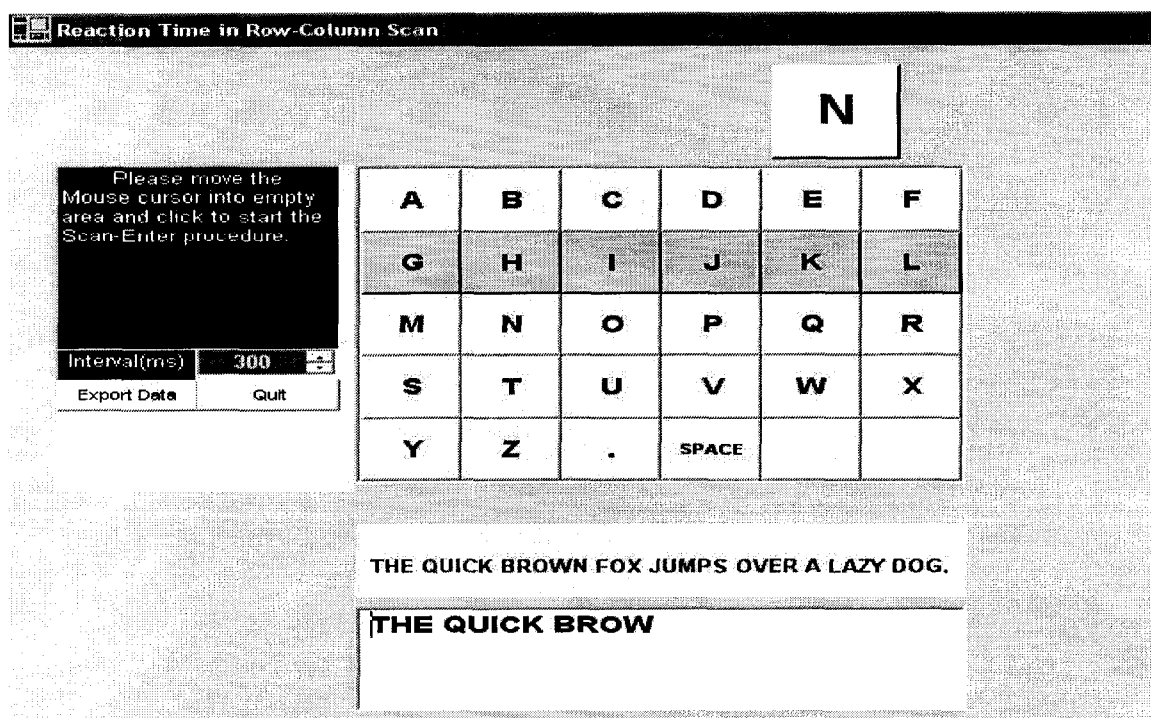


Figure 3.1 Screenshot of Row-Column Interface

While this program was running, another laptop computer was recording force data via an A201 FlexiForce sensor attached to an excitation circuit and a

National Instruments DAQ system. The FlexiForce sensor and excitation circuit are shown in Figure 3.2. The National instruments DAQ system consisted of a National Instruments SCXI-1000DC 4-slot DC-powered chassis which was powered by a National Instruments SCXI-1382 Battery Pack. The battery pack made this signal conditioning unit a portable device. The chassis housed a National Instruments SCXI-1303 Terminal Block that took the output from the FlexiForce circuit and conditioned the voltage signal by converting the analog signal to a digital signal that could be interpreted by the computer. This data was then fed into a laptop computer via a National Instruments 6036 DAQCard for PCMCIA.

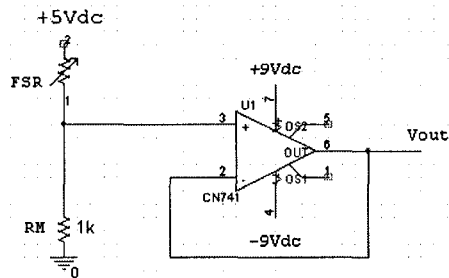
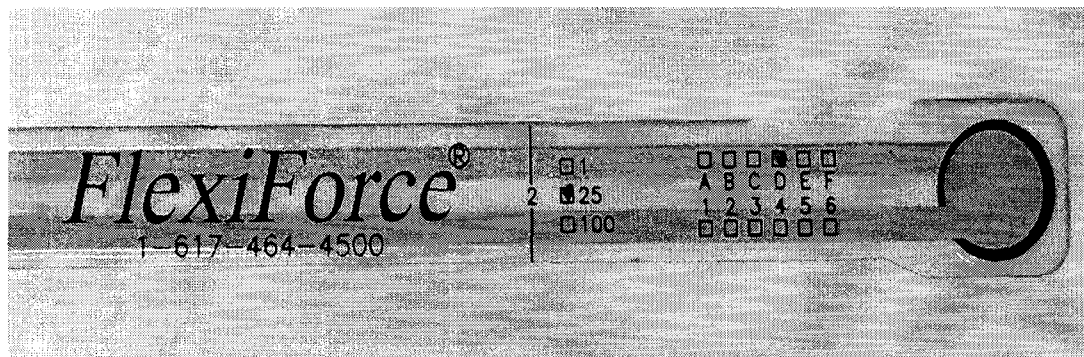


Figure 3.2 Flexiforce Sensor and Excitation Circuit Used to Gather Force Data[18]

Figure 3.3 is an image of the signal conditioning unit. The signal conditioning unit was controlled by a LabVIEW Virtual Instrument which gathered voltage samples at a rate of 25 samples per second.

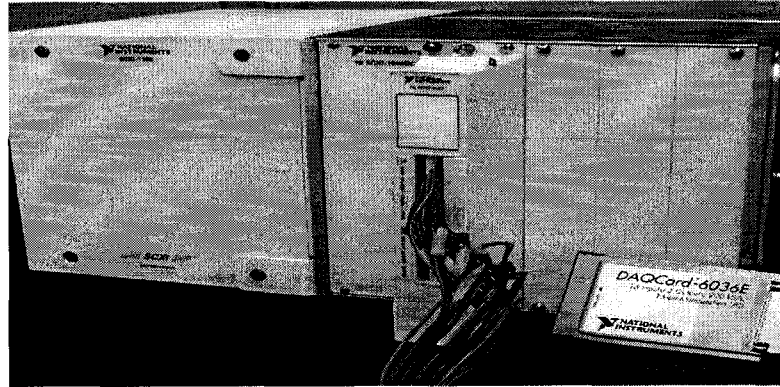


Figure 3.3 National Instruments SCXI-1000DC and 6036E DAQCard[42]

The voltage data was saved to a file for later analysis. In order to get force data from the voltage data, the FlexiForce sensor had to be calibrated. This calibration was conducted by placing a known mass on the force sensor, and recording the voltage using the LabVIEW program. This process was repeated using multiple weights and a linear equation was found relating voltage to force:

$$F = 1.12 \cdot V - 2.24 \quad (1)$$

Because this experiment involved human subjects, a letter was submitted to the Human Use Committee for permission to test using human subjects. The Human Use Committee functions as the Institutional Review Board (IRB) for Louisiana Tech University. Once the request for permission was approved, the experiments were conducted. For this first set of experiments, eight individuals

without a disability participated in the tests. This group was made up of college students, consisting of five males and three females ranging in age from 19 to 23. The individuals ran through a battery of tests which took approximately two hours for each participant. In order to compensate the students for their time, these individuals were all given a bonus point for their grade in Dr. Stan Cronk's class.

This set of tests involved using the scanning interface shown in Figure 3.1 to spell the sentence "THE QUICK BROWN FOX JUMPS OVER A LAZY DOG." In order to minimize any learning effects, the participants were asked to run through the test multiple times before any data was collected. A pilot study conducted earlier showed that after one or two runs, no learning effect could be seen in the data collected. The user started the test with a scan delay of 800ms, and after successful completion of the test the participant was asked to describe the quickness of the scan delay using one of five possible statements:

- Much Too Slow
- A Little Too Slow
- Just About Right
- A Little Too Fast
- Much Too Fast

After each successful spelling of the sentence, the scan delay was decreased by 20% and the user was asked to spell the sentence again using the row-column scanning interface. After each run the participant was again asked to describe the

quickness of the scan delay using one of the five possible statements. The set of experiments was stopped once the user felt the speed made the interface too difficult to operate successfully, or until he or she reached a scan delay of 108ms.

Throughout the battery of tests, data was recorded from both the scanning interface itself, and the LabVIEW program. In order to synchronize both sets of data, a MATLAB program was written to find each voltage peak that coincided with a switch activation timestamp found in the scanning interface data. Because the data from the sensor was not smooth, it was difficult to determine which voltage peaks were switch activations, or simply signal noise. It was necessary to filter the data using a moving average filter. The timestamps on this filtered data was then compared to the timestamps contained in the Excel data file, and an algorithm was written in MATLAB to match the voltage peaks to the switch activations. Once the correct timestamp was determined for the filtered LabVIEW data, the unfiltered data was then used to find the exact voltage output from the sensor. This voltage was then input into the calibration equation and a force was determined for each switch activation.

3.2 Linear, Row-Column, and Overscan Experiments

The second set of experiments involved testing three separate scanning techniques. Two scanning techniques currently employed by clinicians, linear and row-column scanning, were investigated as well as the overscan technique. These tests were the first academic investigation into the overscan technique, and

the goal of these tests was to compare the overscan technique to the two other techniques used by many individuals with physical disabilities. This group of tests gathered the same data as the initial tests involving only row-column scanning, and employed the same testing apparatus as well.

For these tests, a separate IRB proposal was written to request the approval of the testing. Once approval was granted by the IRB committee, ten individuals without a disability were recruited to participate in these tests. This group was made up of college students, consisting of six males and four females ranging in age from 19 to 26. All individuals were presented with an IRB approved consent form, and these individuals were compensated with two extra credit points on their final grade in Dr. Cronk's undergraduate class.

In order to conduct a more accurate simulation of persons with disabilities using a scanning interface, the control switch for these tests was changed from the laptop switch to a membrane switch. This set of experiments was conducted using the Don Johnston Switch Interface Pro shown in Figure 3.4, connected to a Tash Membrane Switch shown in Figure 3.5. The membrane switch is typically plugged into a AAC device using the standard 3.5mm jack plug. The Switch Interface Pro allows the switch to be used by taking the input from the switch, and converting the signal to mouse clicks via the USB port of the computer.

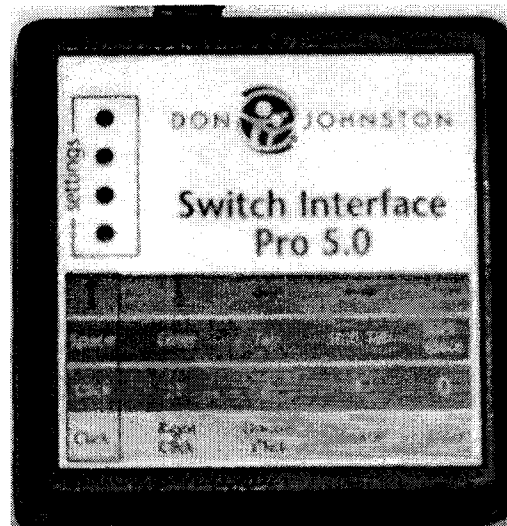


Figure 3.4 Don Johnston Switch Interface Pro 5.0

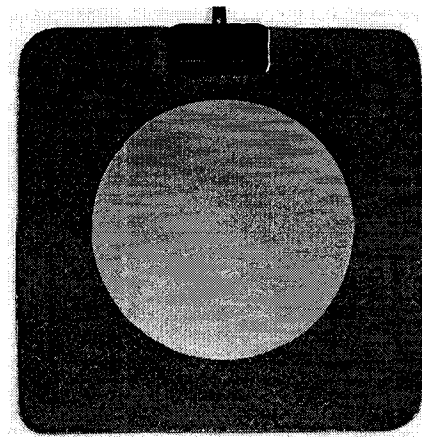


Figure 3.5 Tash Membrane Switch

The larger control switch used in this set of experiments required a larger force sensor than the FlexiForce sensor. For these experiments a force sensing resistor, part number 406 from Interlink Electronics, shown in Figure 3.6, was placed on top of the thin membrane switch to gather force data. This data was sent to the signal conditioning unit, and the LabVIEW system recorded the voltage data at 1000 samples per second.

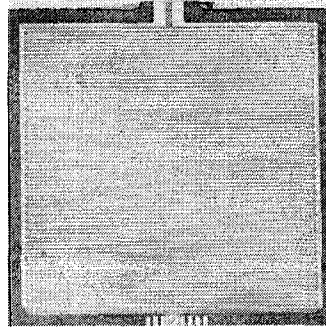


Figure 3.6 Interlink Electronics Force Sensing Resistor

This larger sensor did not behave uniformly when pressure was applied at the edges of the sensor, so a target was drawn on the sensor to instruct participants to hit the sensor in the center. When force was applied to the sensor in the same location, the repeatability of measurements was excellent.

This set of tests was conducted using a scanning interface similar to the interface used in the first set of experiments. The interface for this set of tests was modified to implement linear scanning and overscan as well as row-column scanning. The participant started the scan delay at a speed of 450ms, 300ms or 200ms. These speeds were chosen because users reported that the slower two speeds were in a comfortable range, and the 200 ms was on the edge of the comfortable range. In order to prevent a statistical biasing, three subjects started at 450ms, four started at 300ms, and three started at 200ms. After the linear tests were completed, the participant was tested using the row-column technique. Participants conducted these tests in the same statistically balanced fashion as the linear tests. During these tests, the user was tested at the same scan delays as

the linear test. Finally, the user was tested using the overscan technique. For the overscan technique, the user completed the test at three different forward speeds for each reverse speed, for a total of nine tests. The reverse speeds were the same speeds used in the linear test. The forward speeds were determined by dividing the reverse speeds by three, six, and nine. For example, at the 450ms reverse speed, the users completed the test at a forward speed of 150ms, 75ms, and 50ms.

3.3 Internet Browsing Experiments

The third set of experiments involved the development of an Internet browsing interface which employed the scanning technique to move between the links. This interface was developed in the Visual Basic .NET programming language. It was very important that the Internet interface be integrated into a computer's web browser instead of the alternative, where an entirely new browser is developed. The reasoning behind this was twofold. First, programmers at Microsoft spent a great deal of time developing and testing Internet Explorer. It would be practically impossible to create a web browser under the scope of this project which would perform at the same level as existing technology. By focusing exclusively on the scanning interface, this aspect of the browser was much more developed than it would have been if some of the development time had been devoted to tasks such as handling browser security and interpretation of JavaScript. Second, because the interface was integrated into the web browser, the program continued to work even as new versions of

Internet Explorer were released. It was essential that this interface was tied to the Microsoft framework because Microsoft continues to follow the same basic framework for every new release of Internet Explorer, while adding new functionality. The fact that the Internet scanning interface was tied to this Microsoft framework meant that the interface continued to work as new updates to Internet Explorer were released, even while the development was in progress.

Because it was necessary to integrate the scanning interface into the browser, only programming languages that offer this ability were considered. The scanning Internet interface was developed using Microsoft's Visual Basic product line, the VB.NET programming language. This software was specifically chosen for this project because of the inclusion of two special tools included in the language that made the scanning interface integrate seamlessly into the web browser on a user's computer. These two tools are the Shell Document Object and Control Library (SHDocVw.dll) and Microsoft HyperText Markup Language (MSHTML.dll), which together comprise the WebBrowser Control of the VB .NET programming language. SHDocVw.dll gives the programmer the ability to control navigation, linking, history, favorites, and other aspects of Internet Explorer. MSHTML.dll lets the programmer parse the HTML code on a web page, examining the code for links, anchor elements, images, etc.

The web browsing interface was designed to scan through each link in a HTML document in a similar manner to the earlier scanning interface which highlighted letters instead of links. Each link was highlighted sequentially, and

when the user hit the control switch, the interface received the address to the URL defined by the highlighted link.

Unlike written text, which has an unambiguous sequence of letters that comprise a sentence, the World Wide Web has many paths which can lead to a particular destination. Therefore, it was essential that the individuals all follow a set pattern of links in order to get meaningful data. A test was developed where the user navigated through six web pages in order to complete the task of checking on the standings of the Louisiana Tech Men's Basketball Team. Participants were required to navigate through the same six pages at different scan delay speeds using both the linear and overscan techniques. Each participant ran through the test a total of three times using the same scanning technique and scanning delay, and went through twelve sets of tests for a total of thirty-six total times through the test. Participants used the interface at the same scan delays set in the previous set of tests.

The entire test is represented by the screenshots shown in Figures 3.7 through 3.12. If the participant selected a link which was incorrect, the screen shown in Figure 3.13 was displayed for two seconds and then the user was taken back to the previous page.

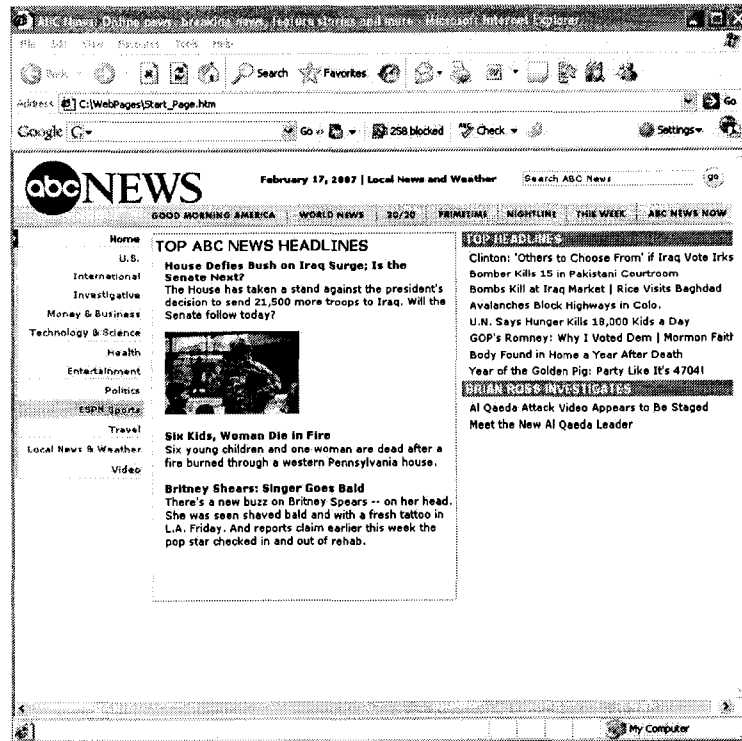


Figure 3.7 First Webpage of Internet Browsing Test

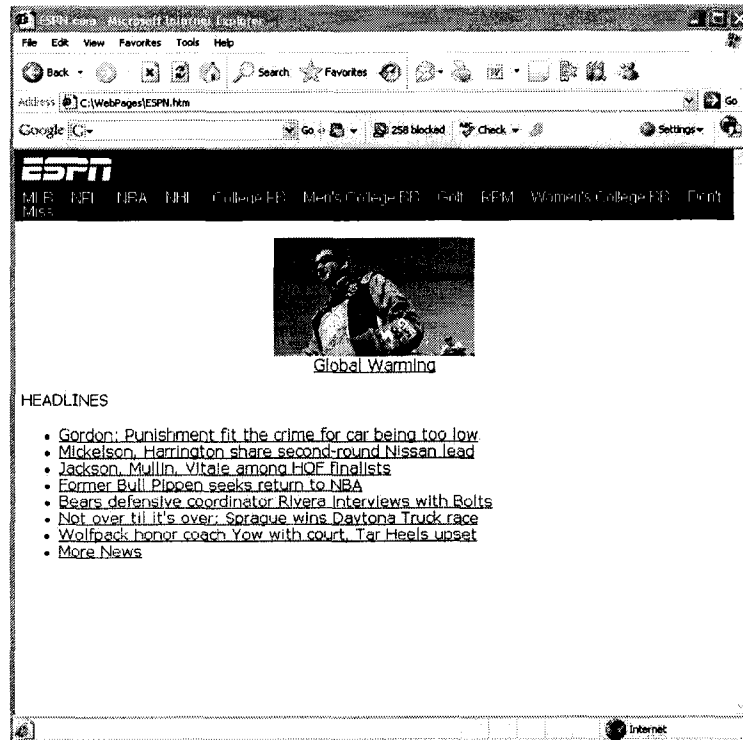


Figure 3.8 Second Webpage of Internet Browsing Test

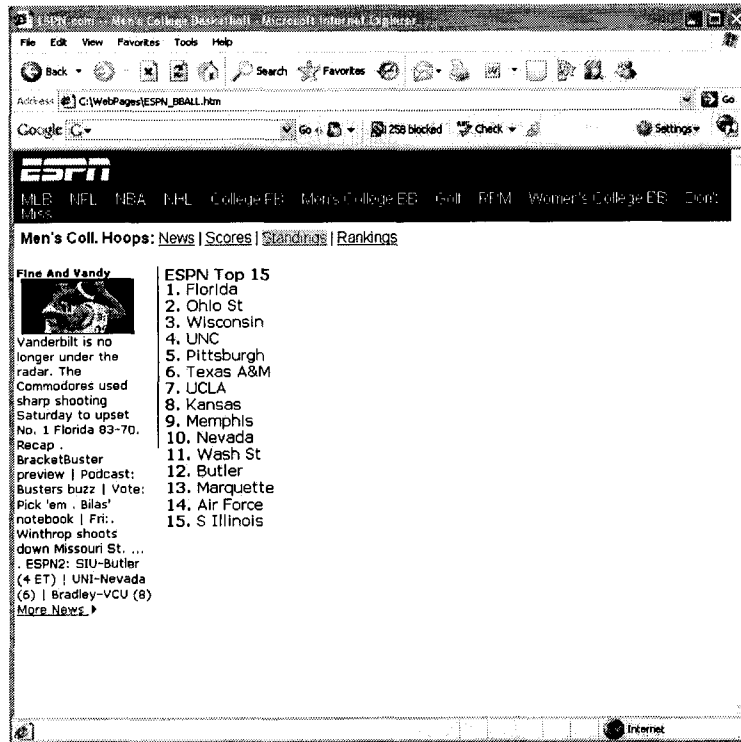
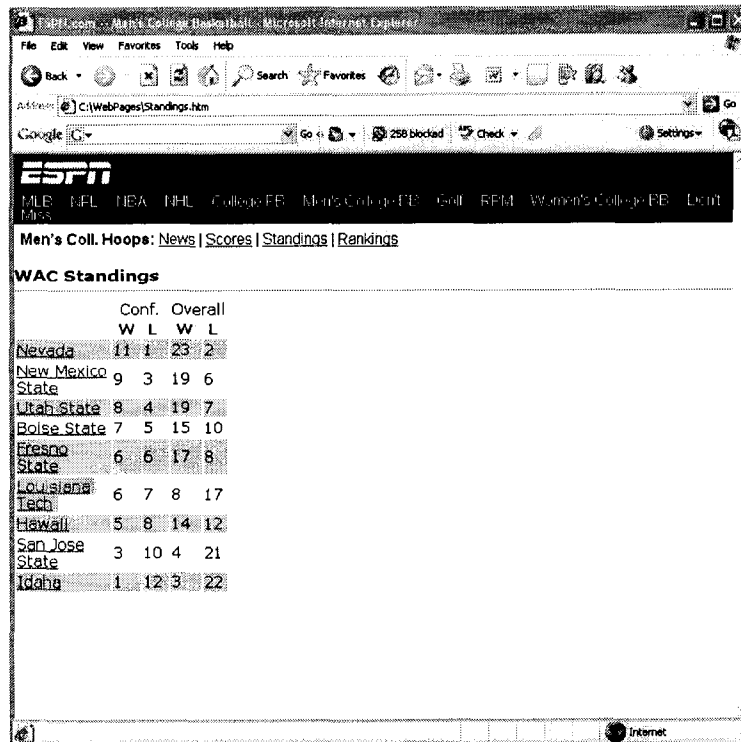


Figure 3.9 Third Webpage of Internet Browsing Test



Figure 3.10 Fourth Webpage of Internet Browsing Test



ESPN

MLB NFL NBA NHL College FB Men's College FB Golf FIM Women's College FB Leat
Misc

Men's Coll. Hoops: [News](#) | [Scores](#) | [Standings](#) | [Rankings](#)

WAC Standings

	Conf.		Overall	
	W	L	W	L
Nevada	11	1	23	2
New Mexico State	9	3	19	6
Utah State	8	4	19	7
Boise State	7	5	15	10
Fresno State	6	6	17	8
Louisiana Tech	6	7	8	17
Hawaii	5	8	14	12
San Jose State	3	10	4	21
Idaho	1	12	3	22

Figure 3.11 Fifth Webpage of Internet Browsing Test

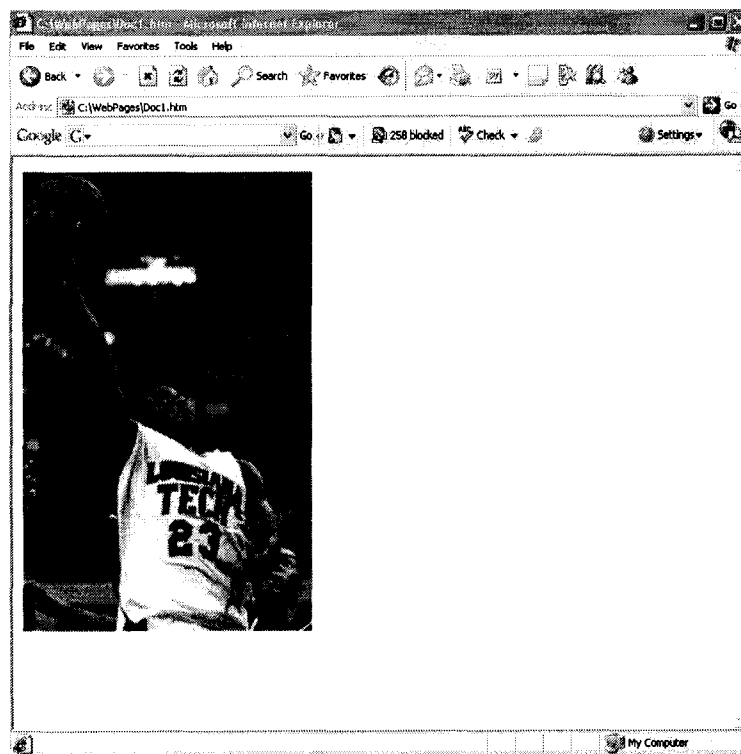


Figure 3.12 Completion Webpage of Internet Browsing Test



Figure 3.13 Missed Link Page Internet Browsing Test

Each webpage used in the test was downloaded from the Internet and cached on the hard drive of the computer running the interface. This was done because testing took place over the course of three months, and the pages online would change many times during the testing period. The caching of pages also served to avoid any latency issues involved with using the network which would taint any timing data.

Once the interface was developed and sufficiently tested, a separate IRB proposal was written to request the approval of the testing. Once approval was granted by the IRB committee, 30 individuals without a disability were recruited to participate in these tests. This group was made up of college students ranging

in age from 19 to 28. All individuals were presented with an IRB approved consent form, and these individuals were compensated with two extra credit points on their final grade in Dr. Cronk's undergraduate class.

3.4 GOMS Modeling

In clinical practice the error rates and reaction times of the individual using the interface are what will determine the communication rate for each individual user. For this reason three GOMS models have been developed to determine how changes in these variables will affect the communication rates of individuals using the various scanning techniques. These three models were developed using the rules of GOMS, using test parameters without using actual test data. Equation 2 is a GOMS model of the linear scan implemented in this dissertation.

$$T = P + C \cdot (1 + E) \cdot ((TP - 1)SD + RT + S) \quad (2)$$

T = Total Completion Time

P = Preparation Time to Start Scan

C = Number of Characters/Links

E = Errors per Character/Link

TP = Position of the Target Character/Link

SD = Scan Delay

RT = Reaction Time

S = System Response Time

System Response Time, Number of Characters, Position of the Target Element, and Scan Delay were all set at predefined levels. The average position of the target elements can be calculated using the position of the elements along with a frequency of use table. Table 3.1 was generated using a frequency of use table for all letters of the English language along with an assumption that each word contains 4.5 characters and each sentence contains 10 words[46][39]. Using Table 3.1, along with the positions of each character in the scanning matrix, the target position for the average element was calculated to be 14.69.

Table 3.1 English Language Frequency of Use

Letter	Frequency	Letter	Frequency	Letter	Frequency
a	0.065628	j	0.001229	s	0.050842
b	0.011989	k	0.006204	t	0.072771
c	0.022355	l	0.032344	u	0.022163
d	0.034176	m	0.019334	v	0.007859
e	0.10207	n	0.054233	w	0.018964
f	0.017904	o	0.060324	x	0.001205
g	0.016192	p	0.015501	y	0.015863
h	0.04897	q	0.000763	z	0.000595
i	0.055977	r	0.04811	Space	0.178571
				Period	0.017857

Similar GOMS models were developed for the row-column and overscan interfaces. Equation 3 is the GOMS model of the row-column scanning interface. For Equation 3, the average row and column of the target element was calculated using Table 3.1. The average row of the target character is 2.93 and the average column of the target character is 3.11. Equation 4 is the GOMS Model of the overscan interface. Examination of all three equations reveals that all three techniques share some elements, and the difference in completion times can be computed by looking at the elements of the equations which are not common to all three equations.

$$T = P + C \cdot (1 + E) \cdot ((TR + TC - 2)SD + 2RT + 2S) \quad (3)$$

T = Total Completion Time

P = Preparation Time to Start Scan

C = Number of Characters

E = Errors per Character

TR = Mean Row of the Target Element

TC = Mean Column of the Target Element

SD = Scan Delay

RT = Mean Reaction Time

S = System Response Time

$$T = P + C \cdot (1 + E) \cdot ((TP + PP - 1) \cdot FSD + FRT + (PP - 1) \cdot BSD + BRT + 2S) \quad (4)$$

T = Total Completion Time

P = Preparation Time to Start Scan

C = Number of Characters/Links

E = Errors per Character/Links

TP = Mean Position of the Target Character/Link

PP = Mean Positions Past Target Character/Link of
Actual Character/Link

FSD = Forward Scan Delay

FRT = Mean Forward Reaction Time

BSD = Backward Scan Delay

BRT = Mean Backward Reaction Time

S = System Response Time

CHAPTER 4

RESULTS

4.1 Row-Column Scanning

Forces were measured for each user, and Figure 4.1 shows the force profile from a single experimental run. Each peak represents a switch activation, while the resting force is approximately 25 grams for this user. Forces at the beginning of the test tended to be higher than the forces at the end of the test. Figure 4.2 shows the force profile for a single switch activation from the experimental run shown in Figure 4.1. Figure 4.3 is a graphical comparison between the mean force for one user for the first 10 activations of a test versus the mean force used for the last 10 activations. Figure 4.4 is the same comparison for all tests conducted. Because the activations at the beginning of the test were executed with more force, it is hypothesized that users utilize more force when they are not familiar with the scan delay, and as they become more accustomed to the speed the comfort level increases and force decreases.

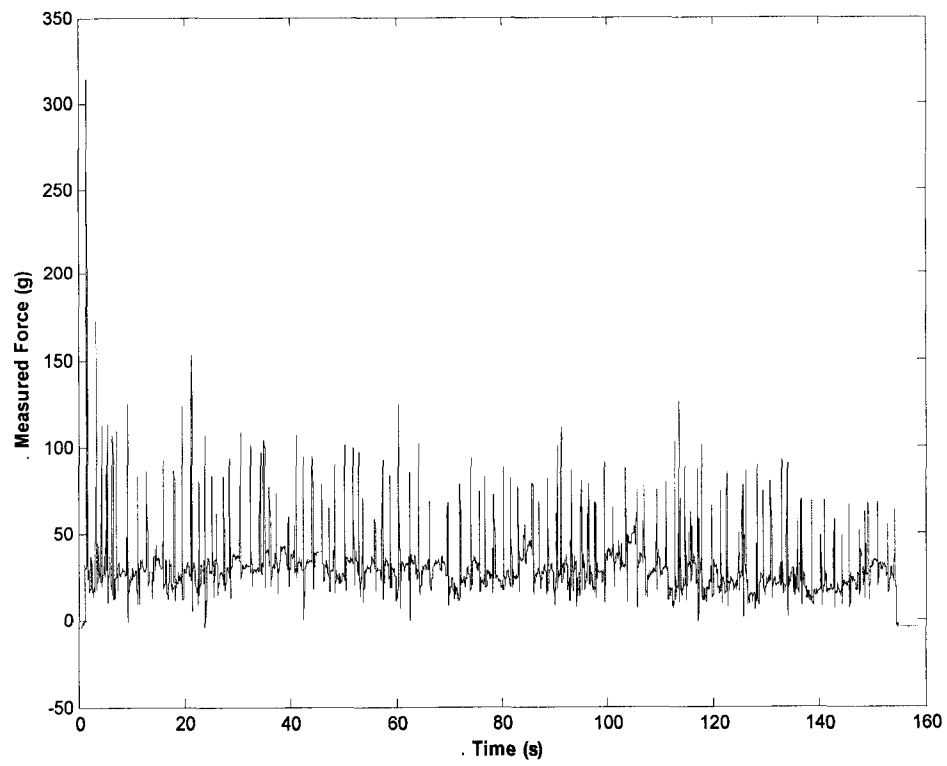


Figure 4.1 Force measurements for single experimental run

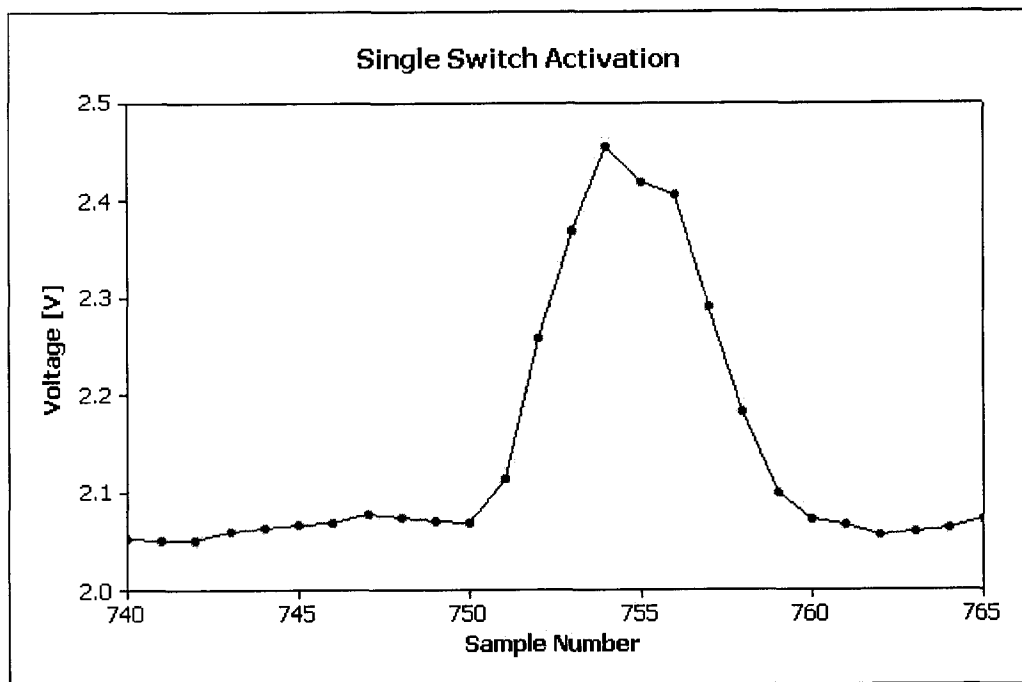


Figure 4.2 Force Profile for a Single Switch Activation.

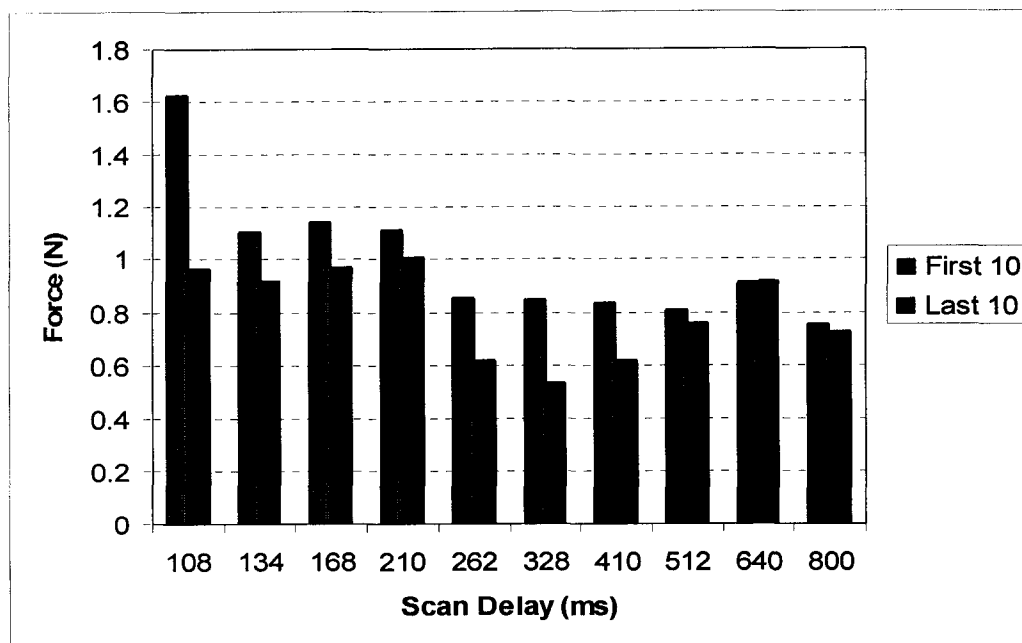


Figure 4.3 Force of First 10 Activations vs. Last 10 Activations for One Participant

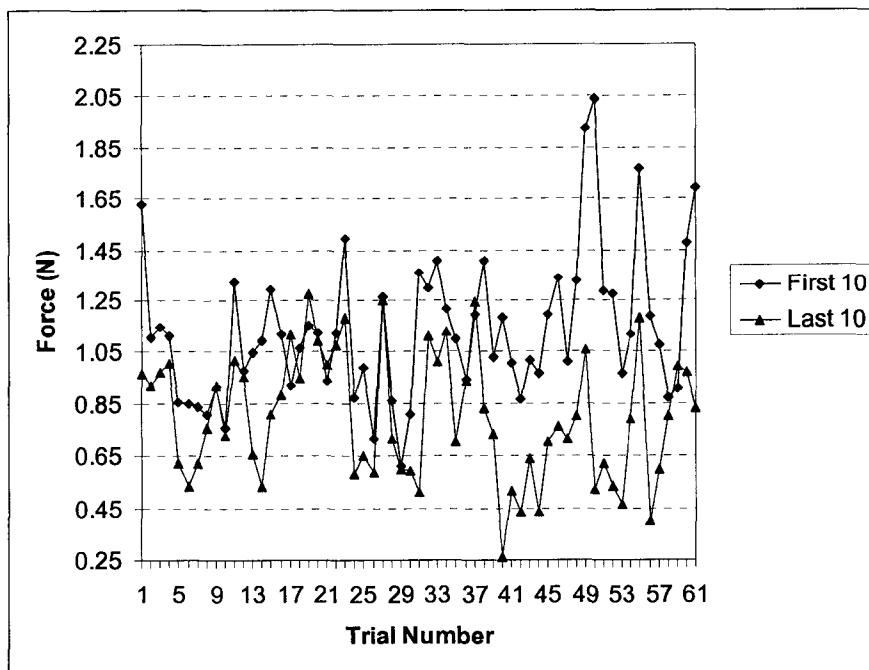


Figure 4.4 Force of First 10 Activations vs. Last 10 Activations for Each Test

The users for the first test were asked to describe the speed of the test with one of five simple phrases. Figure 4.5 shows the speed that each user chose for the “just about right” and “much too fast” speeds. Figure 4.6 shows that the error rate increases dramatically when the user feels the scan delay at a speed that is “much too fast.” This range was later used to design the scan delay range for the second and third experiments. Because each scanning test takes a long time, it was essential to test in the meaningful range so time was not wasted testing inappropriate scan delays.

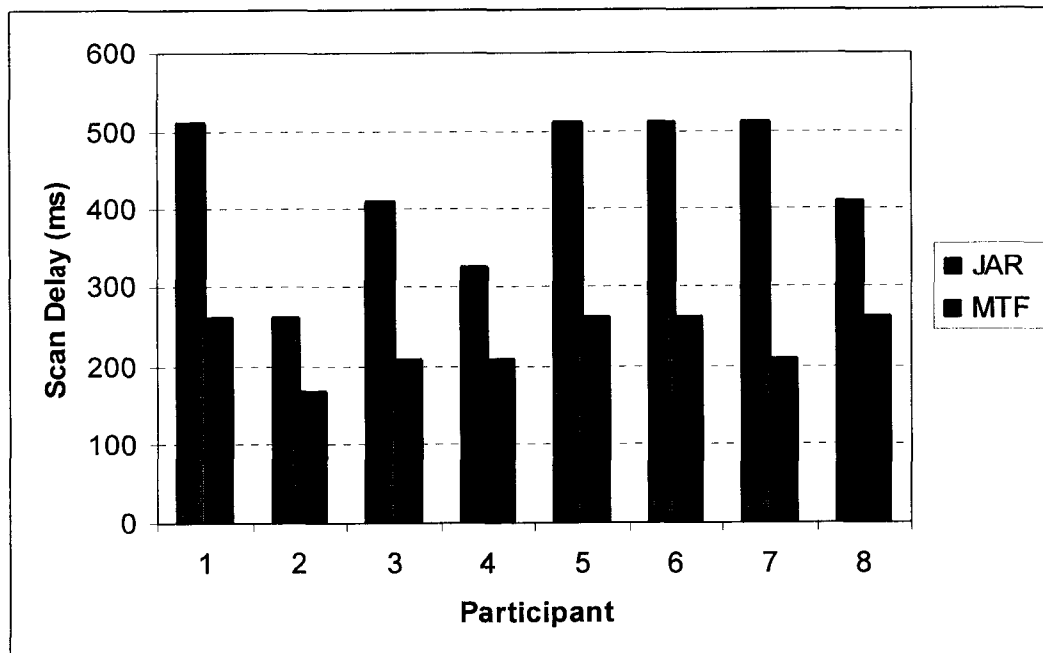


Figure 4.5 Scan Delays deemed "Just About Right" and "Much Too Fast"

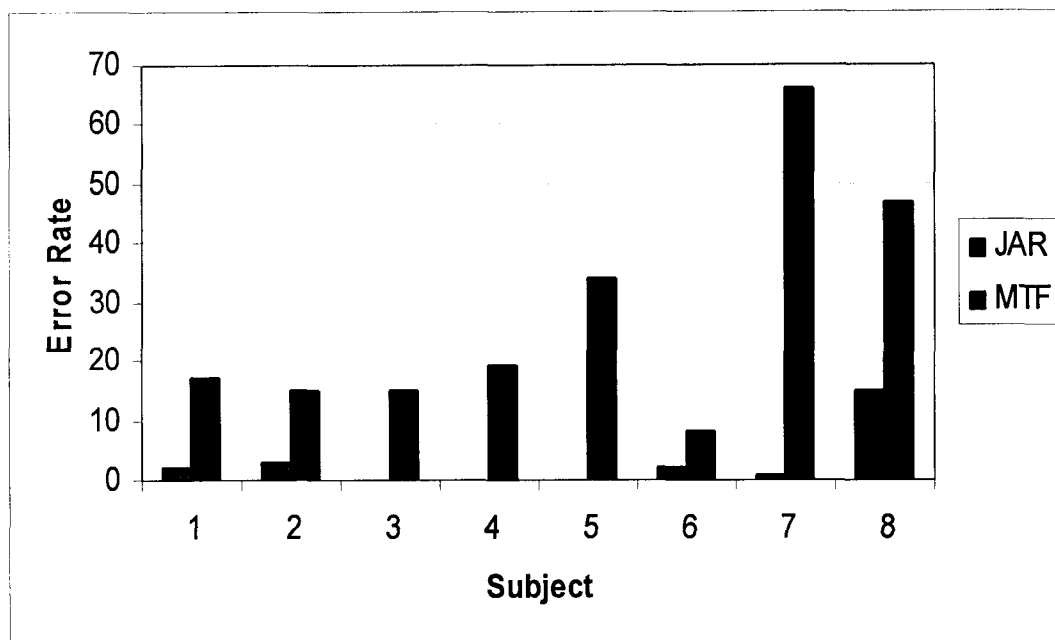


Figure 4.6 Number of errors committed by each subject at scan delays deemed "Just About Right" (JAR) and "Much Too Fast" (MTF).

4.2 Linear, Row-Column, and Overscan Alphabetic Interface

4.2.1 Linear Alphabetic Interface

Figure 4.7 shows the relationship between completion time and scan delay for the linear tests of the alphabetic scan. Users completed the task in a mean time of 343.8 s ($s = 16.3$) at a scan delay of 450 ms. The average completion time dropped to 229.9 s ($s = 10.0$) for a scan delay of 300 ms, and 177.4 s ($s = 16.3$) for a scan delay of 200 ms. Decreasing the scan delay is one method of increasing the scanning communication rate, however if the error rate increases this increase may offset the gains made by decreasing the scan delay.

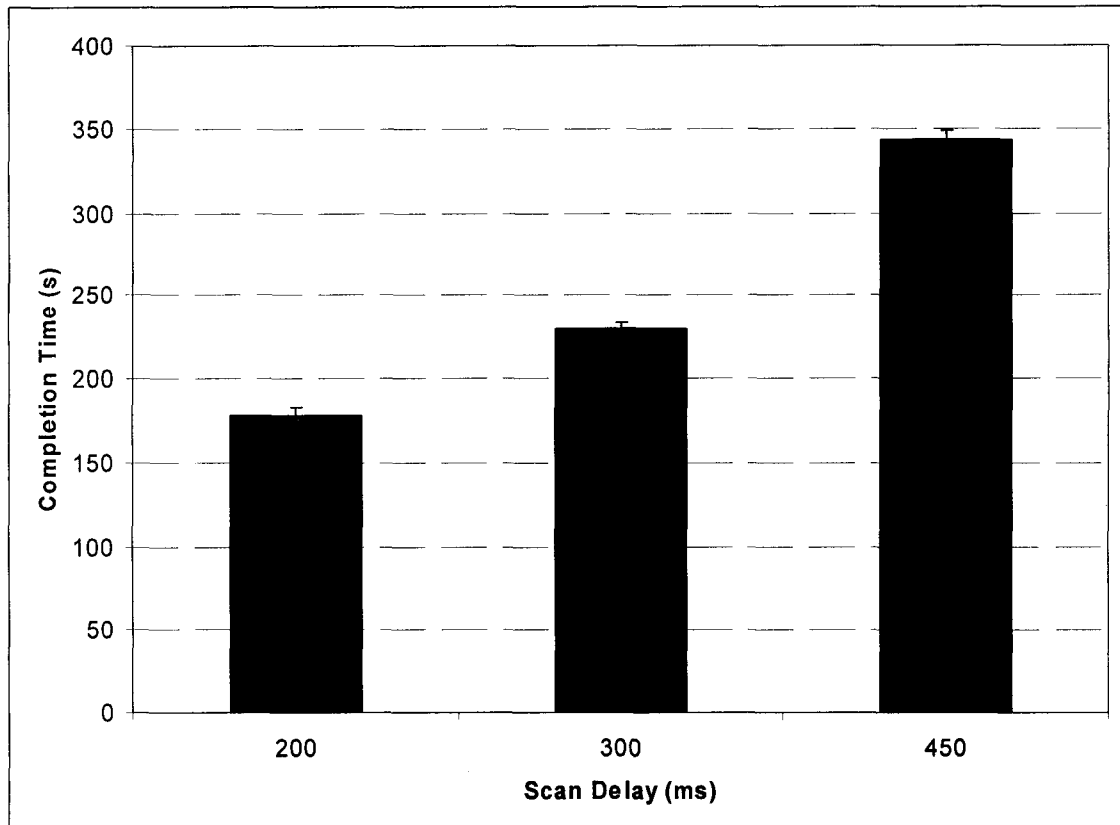


Figure 4.7 Completion Time for All Linear Tests of Alphabetic Scan

The average force in Newtons at the three scan delays is shown in Figure 4.8. The mean force increased slightly as the scan delay was decreased, but the standard error of the means, displayed in the figure as the error bars, overlaps for each of the measured forces. The mean force at 450 ms was 2.86 N ($s = 0.38$), the force at 300 ms was 2.94 N ($s = 0.60$), and the force at 200 ms was 3.27 N ($s = 1.06$).

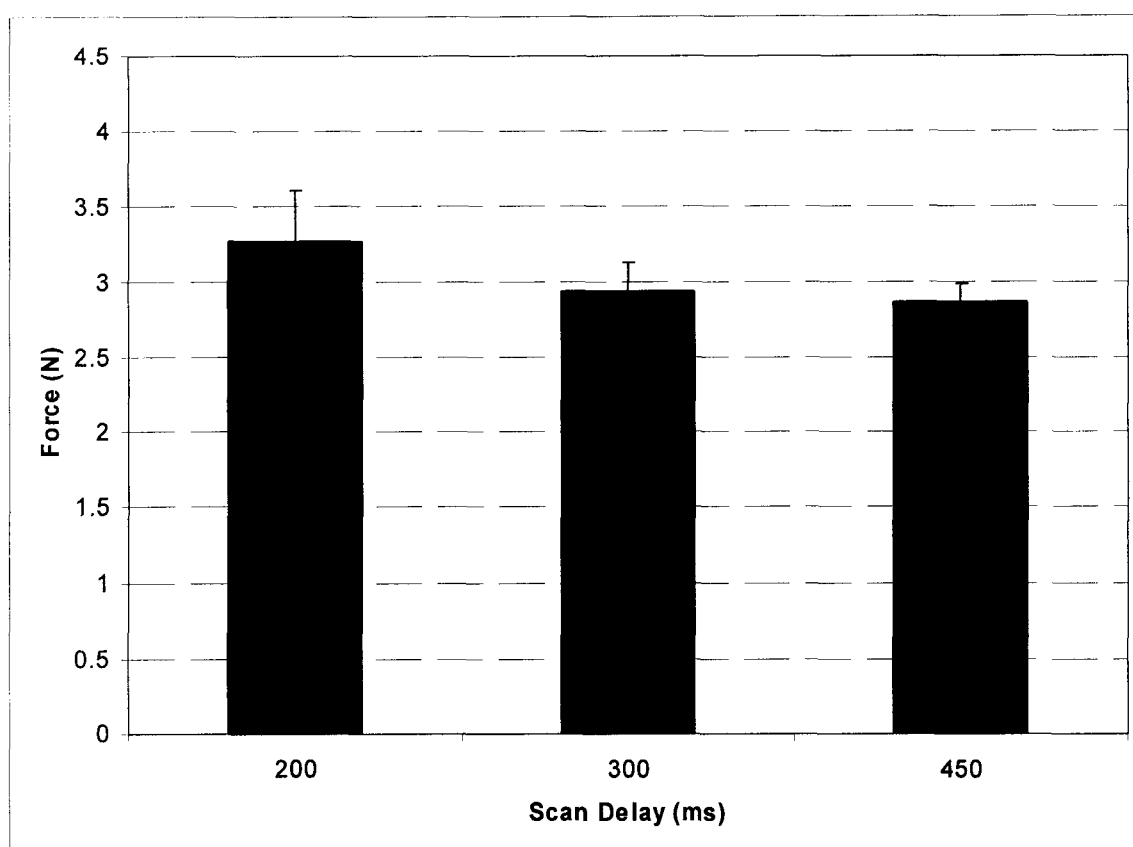


Figure 4.8 Mean Force for All Linear Tests of Alphabetic Scan

Figure 4.9 shows the relationship of key press time to scan delay. The key press time did not change significantly as the scan delay changed, and all means fell well within the standard errors of the two means. For a scan delay of 450 ms the mean key press time was measured at 197.4 ms ($s = 121.2$), at 300 ms the mean key press time was 202.9 ms ($s = 146.2$), and at 200 ms the mean key press time was 194.5 ms ($s = 128.4$).

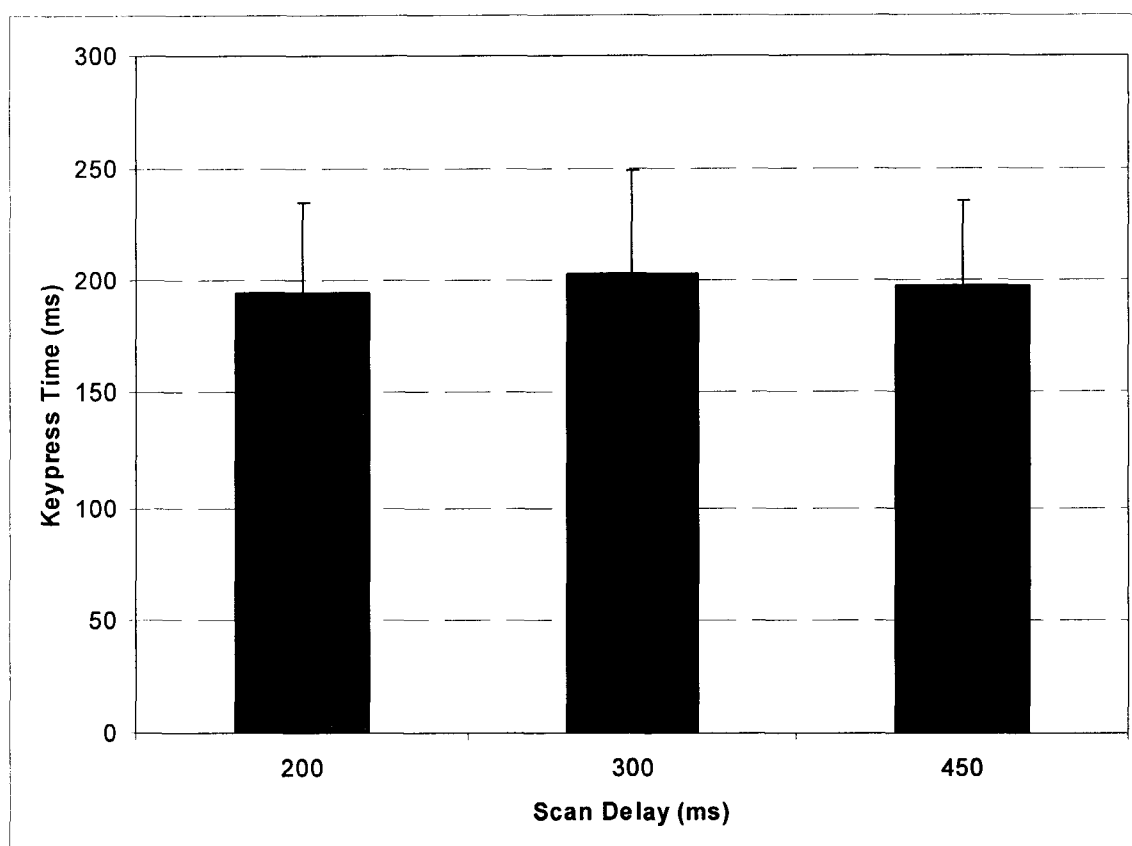


Figure 4.9 Key Press Time for All Linear Tests of Alphabetic Scan

The reaction times decreased as the scan delay decreased, as shown in Figure 4.10. The mean reaction time at a scan delay of 450 ms was 184.4 ms ($s = 39.2$), the reaction time at a scan delay of 300 ms was 148.4 ms ($s = 30.7$), and the reaction time at a scan delay of 200 ms was 109.0 ms ($s = 16.1$). The ratio of reaction time to scan delay increased as the scan delay decreased. This increasing ratio means that users were hitting the switch relatively later in the scan delay interval when the desired selection was highlighted.

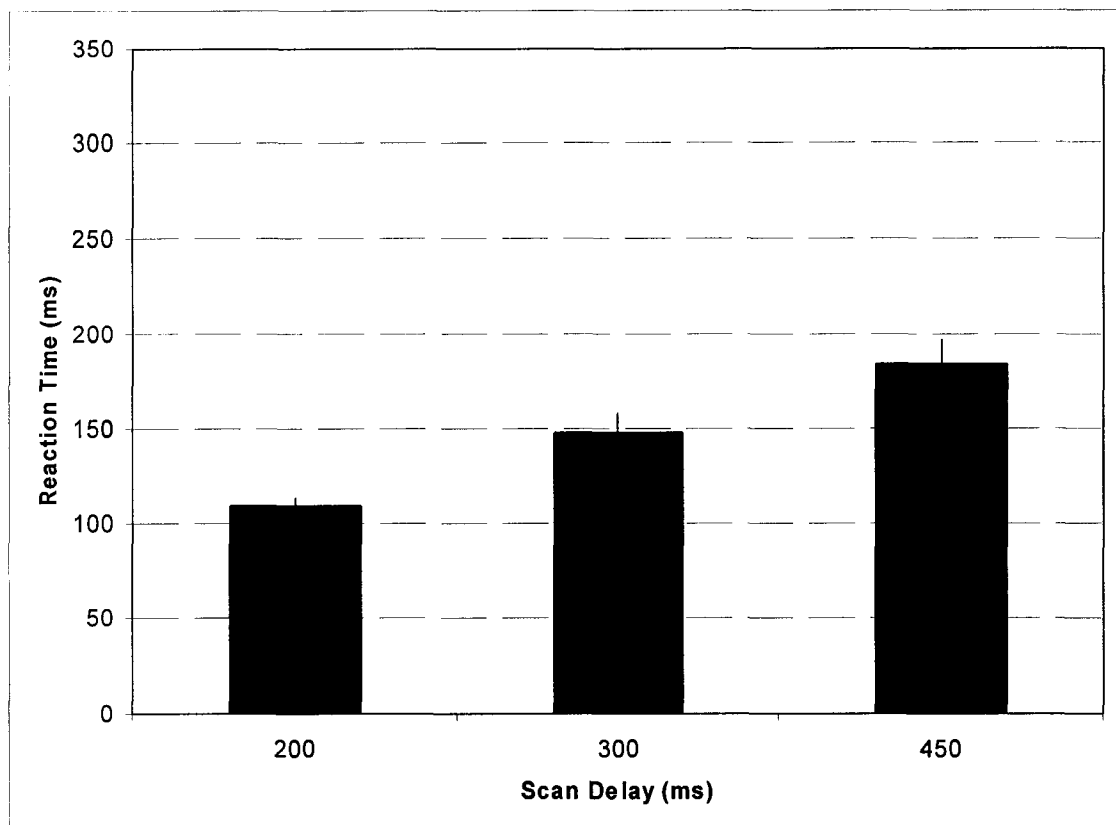


Figure 4.10 Mean Reaction Time for All Linear Tests of Alphabetic Scan

The error rates for the linear scanning tests were similar for scan delays of 450 ms and 300 ms, as shown in Figure 4.11. The error rate for 450 ms was 3.2 ($s = 1.9$) errors per test, and the error rate for a scan delay of 300 ms was 3.1 ($s = 2.1$) errors per test. However, error rates at a scan delay of 200 ms were more than twice the error rates for the tests at the higher scan delays. The error rate at a scan delay of 200 ms was 8.3 ($s = 4.6$) errors per test, as shown in Figure 4.11.

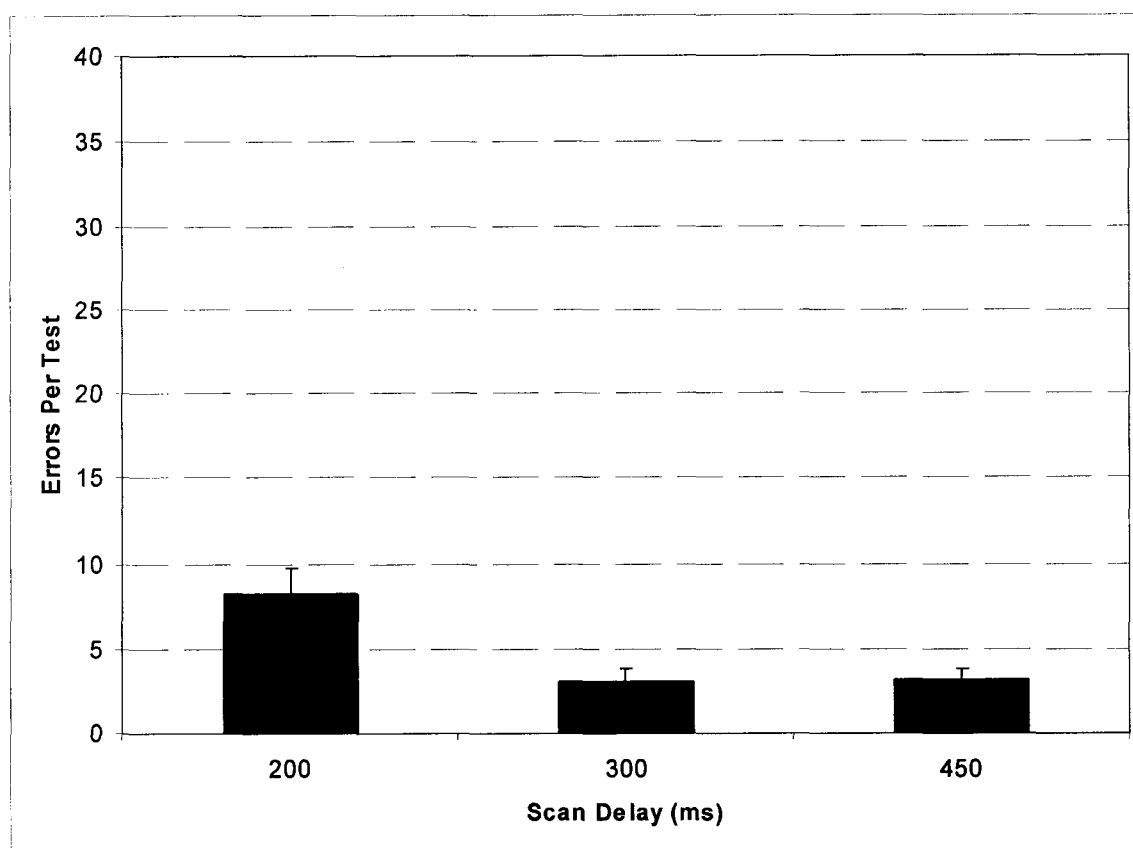


Figure 4.11 Errors vs. Scan Delay for All Linear Tests of Alphabetic Scan

4.2.2 Row-Column Alphabetic Interface

The same sets of data were recorded for the row-column tests. Completion times are shown in Figure 4.12 for all three scan delays of the row-column scanning interface. The mean completion time for the 450 ms scan delay is 159.6 s ($s = 8.9$), the completion time for the 300 ms scan delay is 141.8 s ($s = 15.2$), and the completion time for the 200 ms scan delay is 140.7 s ($s = 34.5$). Completion times for the row-column interface do not improve in the same fashion as the completion times for the linear interface because as the scan delay decreases, the error rates increase at a greater rate. Although users are able to choose the desired selection more quickly with a smaller scan delay, the time savings is negated by the higher error rates.

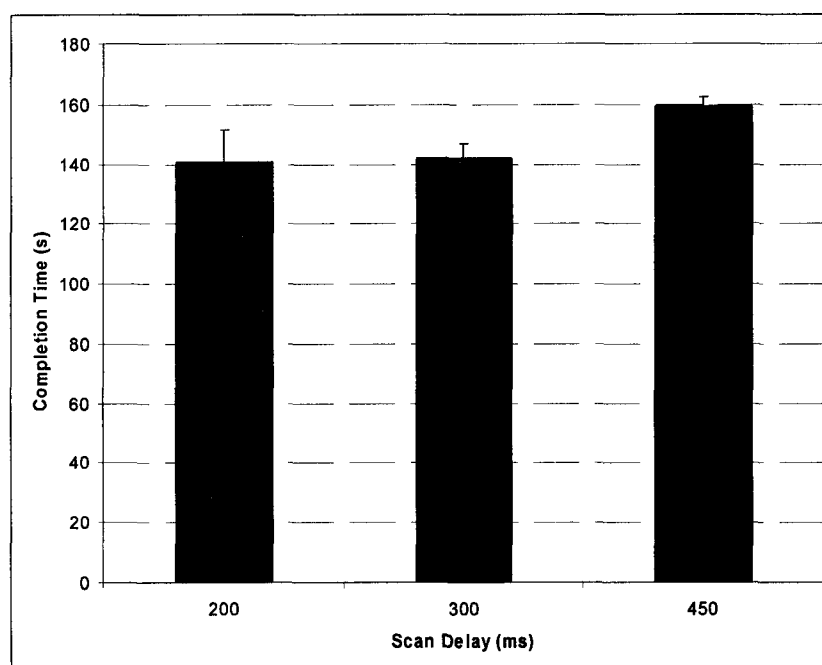


Figure 4.12 Completion Time vs. Scan Delay for All Row-Column Tests of Alphabetic Scan

Figure 4.13 shows a comparison of the mean force at the three different scan delays for the row-column alphabetic scan. At the 450 ms scan delay the mean force was 2.95 N ($s = 0.78$), at the 300 ms scan delay the mean force was 3.22 N ($s = 0.82$), and at the 200 ms scan delay the mean force was 3.85 N ($s = 1.16$). The standard errors of the mean force for the 450 ms scan delay and the 300 ms scan delay overlap, but the 200 ms scan delay has a mean force which is outside the standard error of the mean forces for the other two scan delays.

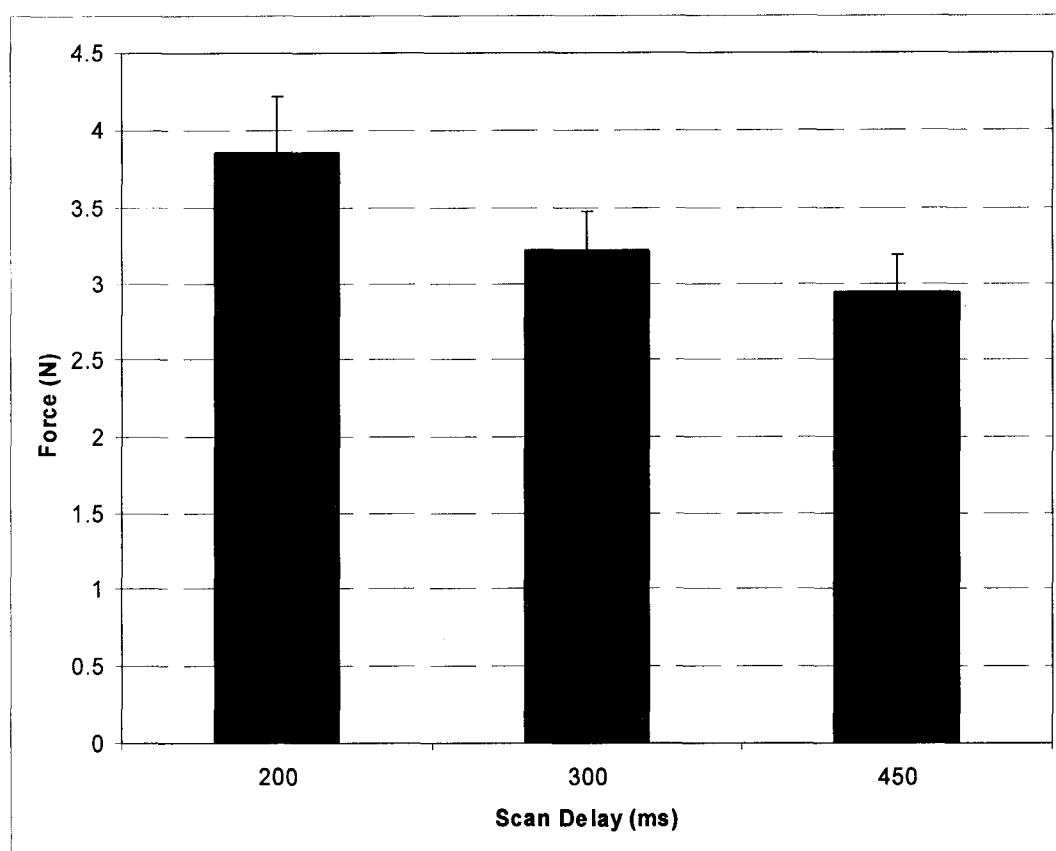


Figure 4.13 Force vs. Scan Delay for All Row-Column Tests of Alphabetic Scan

The key press times for the row-column alphabetic scan, shown in Figure 4.14, do not change significantly as the scan delay is decreased. The standard errors of the mean for all three key press measurements are overlapping. The 450 ms scan delay had a mean key press time of 207.3 ms ($s = 120.0$), the 300 ms scan delay had a mean key press time of 192.2 ms ($s = 117.2$), and the 200 ms key press time had a mean key press time of 192.0 ms ($s = 123.9$).

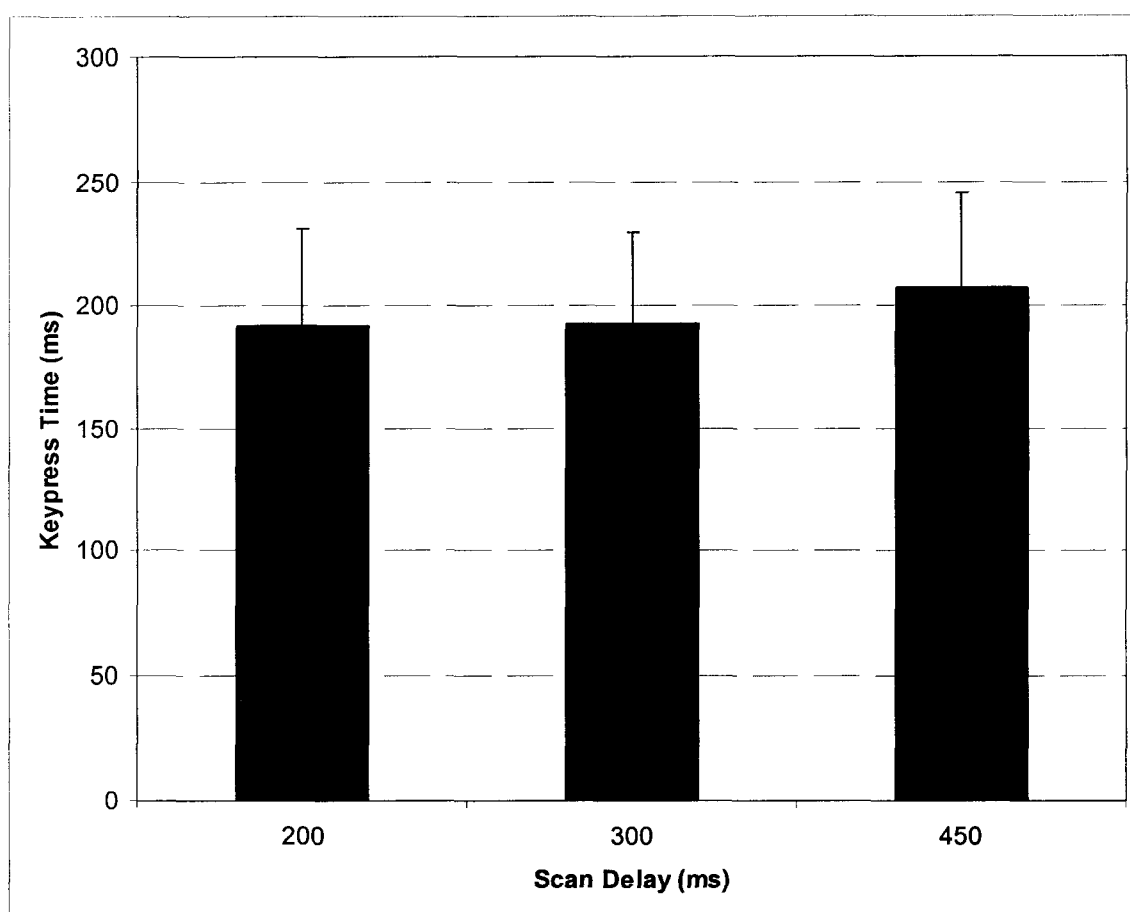


Figure 4.14 Key press Time vs. Scan Delay for All Row-Column Tests of Alphabetic Scan

The reaction times for the alphabetic row-column scan decreased as the scan delay decreased, as shown in Figure 4.15. The 450 ms scan delay had a mean reaction time of 178.8 ms ($s = 41.7$), the 300 ms scan delay had a mean reaction time of 159.4 ms ($s = 30.6$), and the 200 ms scan delay had a mean reaction time of 107.593 ms ($s = 13.6$).

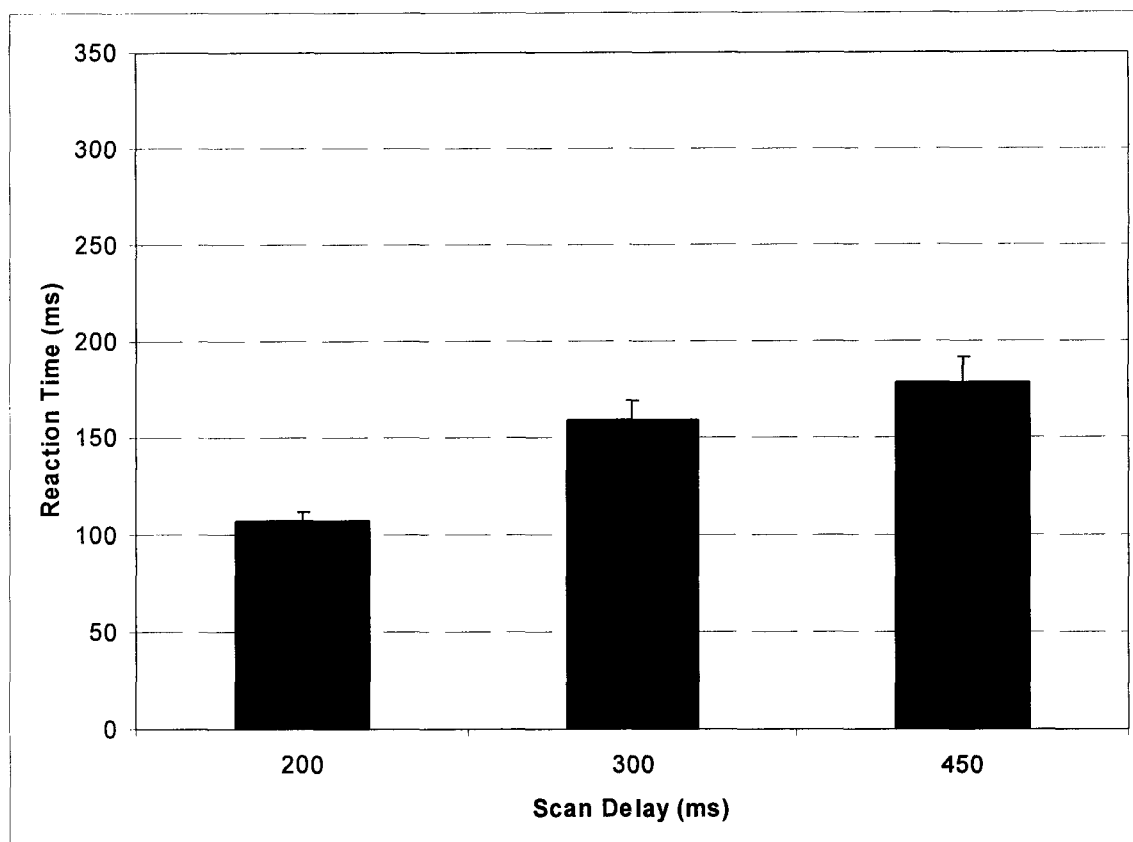


Figure 4.15 Reaction Time vs. Scan Delay for All Row-Column Tests of Alphabetic Scan

The error rate for the row-column alphabetic scan increased dramatically as the scan delay decreased. The mean error rate for the 450 ms scan delay was 6.4 ($s = 1.2$) errors per test, the mean error rate for the 300 ms scan delay was 13.2

($s = 2.5$) errors per test, and the mean error rate for the 200 ms scan delay was 28.7 ($s = 5.2$) errors per test. Figure 4.16 shows the relationship between errors and scan delay for the row-column tests.

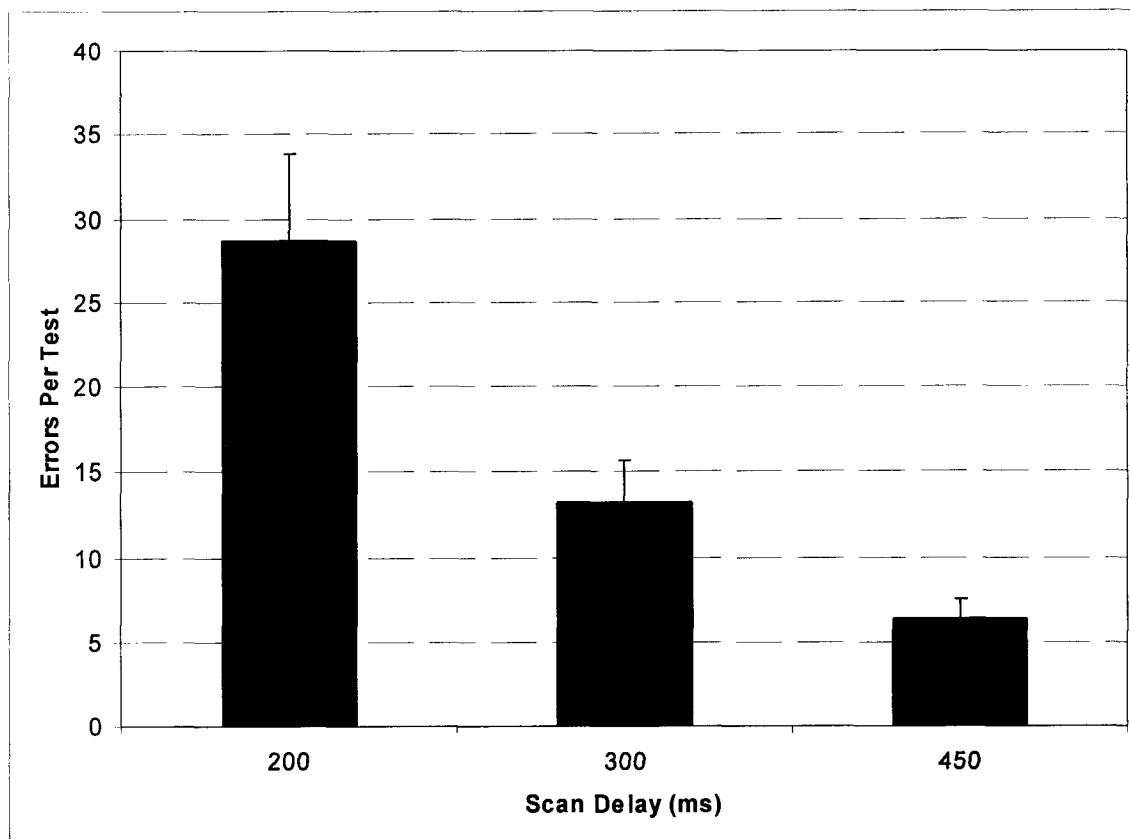


Figure 4.16 Errors vs. Scan Delay for All Row-Column Tests of Alphabetic Scan

4.2.3 Overscan Alphabetic Interface

For backward scan delays of 450 ms and 200 ms, the completion times for scan delays with a forward speed three times faster than the backward speed were significantly different than the completion times for scan delays with a forward speed six or nine times faster than the backward speed. The completion

time for a backward scan delay of 300 ms did not follow this same pattern. The mean completion time for the backward scan delay of 450 ms was 201.5 s ($s = 26.6$) for the forward speed divisor of three, 156.2 s ($s = 13.5$) for the forward speed divisor of six, and 159.5 s ($s = 25.5$) for the forward speed divisor of nine. The mean completion time for the backward scan delay of 300 ms was 151.7 s ($s = 12.5$) for the forward speed divisor of three, 148.7 s ($s = 25.3$) for the forward speed divisor of six, and 146.4 s ($s = 14.8$) for the forward speed divisor of nine. The mean completion time for the backward scan delay of 200 ms was 175.8 s ($s = 36.7$) for the forward speed divisor of three, 143.7 s ($s = 24.0$) for the forward speed divisor of six, and 142.2 s ($s = 28.1$) for the forward speed divisor of nine. Figure 4.17 shows the completion times for all scan delays of the alphabetic overscan tests. The scan delay for all overscan figures is shown as the backward scan delay, followed by an underscore and then the forward divisor.

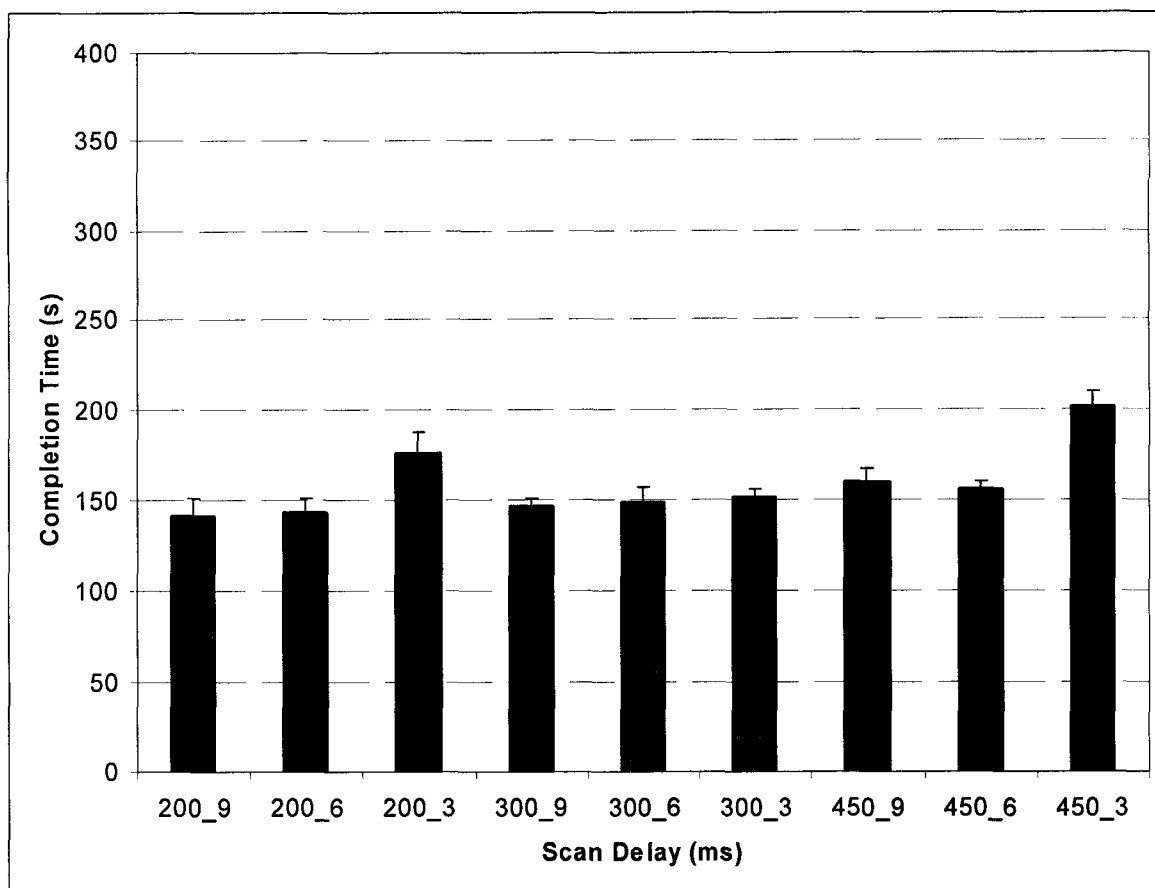


Figure 4.17 Completion Time vs. Scan Delay for All Overscan Tests of Alphabetic Scan

The mean forces measured for the alphabetic overscan tests trended downward with an increasing scan delay, but the standard errors of the means for most of the force measurements overlap. The mean force for the backward scan delay of 450 ms was 3.33 N ($s = 1.02$) for the forward speed divisor of three, 3.67 N ($s = 1.24$) for the forward speed divisor of six, and 3.40 N ($s = 0.95$) for the forward speed divisor of nine. The mean force for the backward scan delay of 300 ms was 3.27 N ($s = 0.94$) for the forward speed divisor of three, 3.50 N ($s = 1.16$) for the forward speed divisor of six, and 3.36 N ($s = 0.77$) for the forward

speed divisor of nine. The mean force for the backward scan delay of 200 ms was 3.71 N ($s = 0.86$) for the forward speed divisor of three, 3.94 N (0.83) for the forward speed divisor of six, and 3.91 N ($s = 1.03$) for the forward speed divisor of nine. The mean forces measured for the alphabetic overscan tests are shown in Figure 4.18

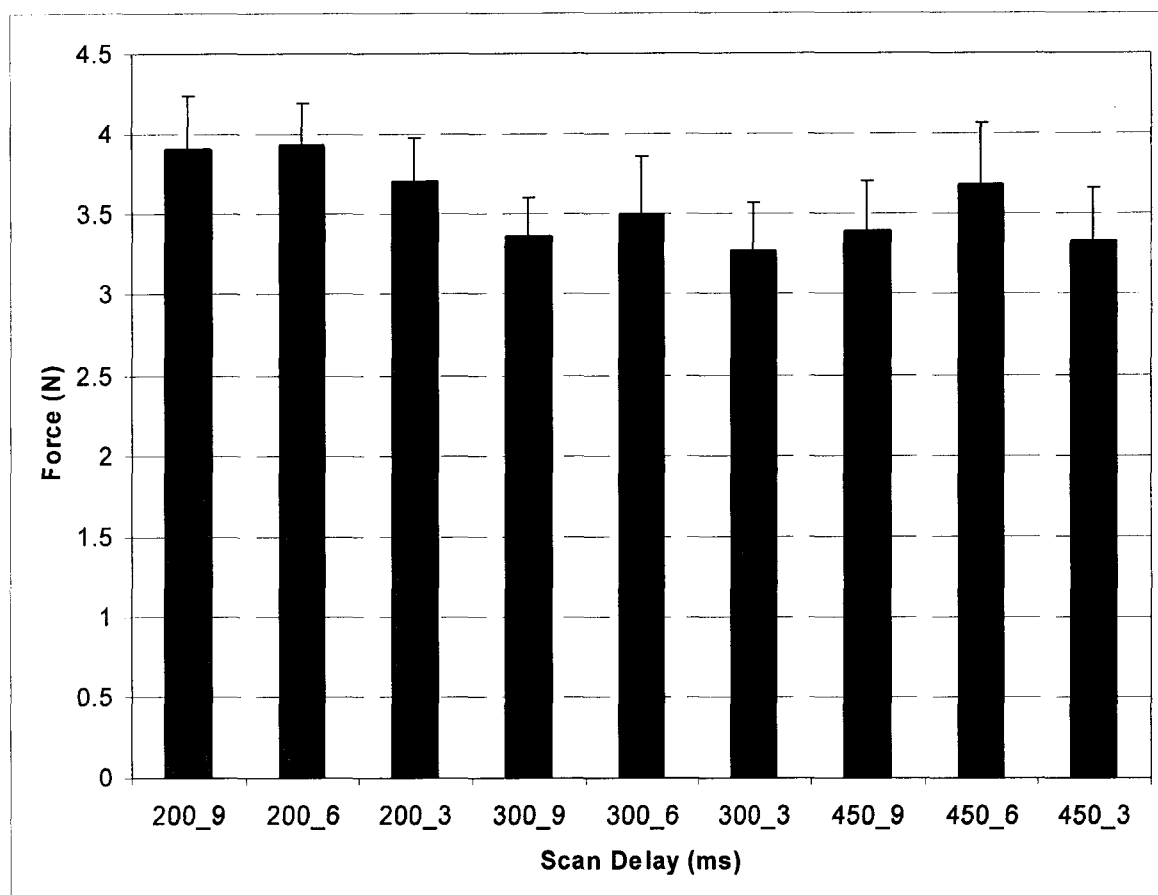


Figure 4.18 Force vs. Scan Delay for All Overscan Tests of Alphabetic Scan

Key press times for the alphabetic overscan tests did not differ statistically for the nine different scan delays. The mean key press time for the backward scan delay of 450 ms was 200.1 ms ($s = 106.7$) for the forward speed divisor of three,

212.8 ms ($s = 124.0$) for the forward speed divisor of six, and 202.0 ms ($s = 115.2$) for the forward speed divisor of nine. The mean key press time for the backward scan delay of 300 ms was 204.0 ms ($s = 109.7$) for the forward speed divisor of three, 199.1 ms ($s = 101.3$) for the forward speed divisor of six, and 218.3 ms ($s = 156.0$) for the forward speed divisor of nine. The mean key press time for the backward scan delay of 200 ms was 211.3 ms ($s = 119.5$) for the forward speed divisor of three, 223.3 ms ($s = 143.1$) for the forward speed divisor of six, and 224.5 ms ($s = 139.3$) for the forward speed divisor of nine. The standard errors of the means for all mean key press times of the alphabetic overscan tests overlap, as shown in Figure 4.19.

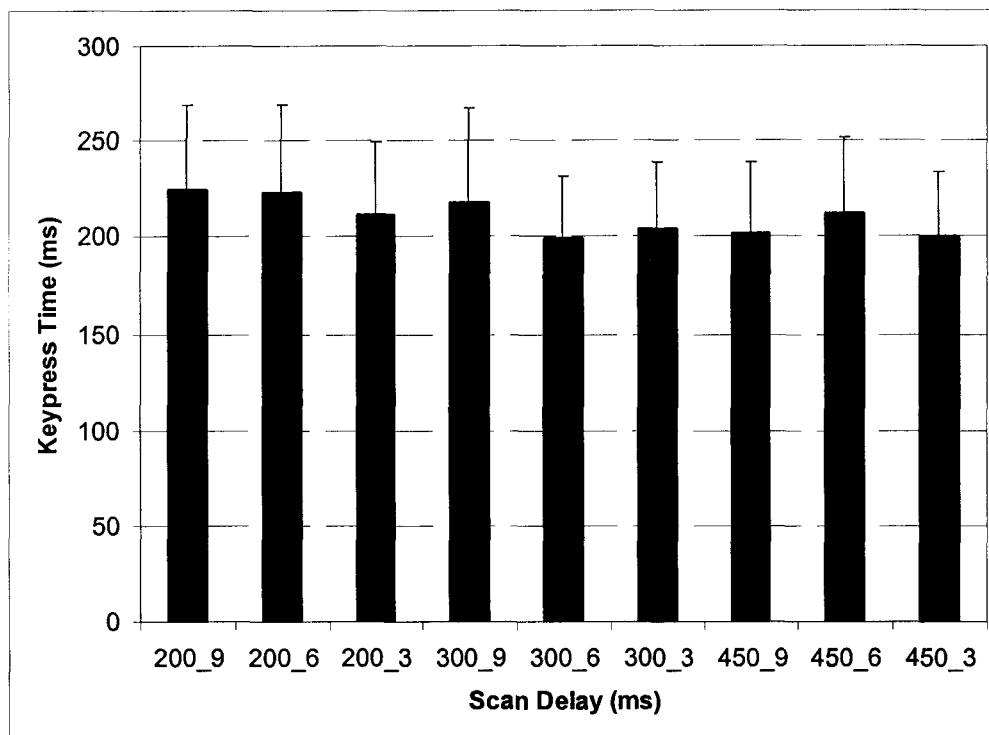


Figure 4.19 Key Press Time vs. Scan Delay for All Overscan Tests of Alphabetic Scan

Figure 4.20 shows the relationship between the forward and backward scan delays and the reaction times measured for the alphabetic overscan tests. Reaction times for the backward scan delay decreased as the forward scan delay was decreased for a given backward scan delay. The mean reaction time for the backward scan delay of 450 ms was 149.5 ms ($s = 35.7$) for the forward scan and 321.0 ms ($s = 66.5$) for the backward scan for the forward speed divisor of three, 212.8 ms ($s = 124.0$) for the forward speed divisor of six, and 202.0 ms ($s = 115.2$) for the forward speed divisor of nine. The mean reaction time for the backward scan delay of 300 ms was 204.0 ms ($s = 109.7$) for the forward speed divisor of three, 199.1 ms ($s = 101.3$) for the forward speed divisor of six, and 218.3 ms ($s = 156.0$) for the forward speed divisor of nine. The mean reaction time for the backward scan delay of 200 ms was 211.3 ms ($s = 119.5$) for the forward speed divisor of three, 223.3 ms ($s = 143.1$) for the forward speed divisor of six, and 224.5 ms ($s = 139.3$) for the forward speed divisor of nine.

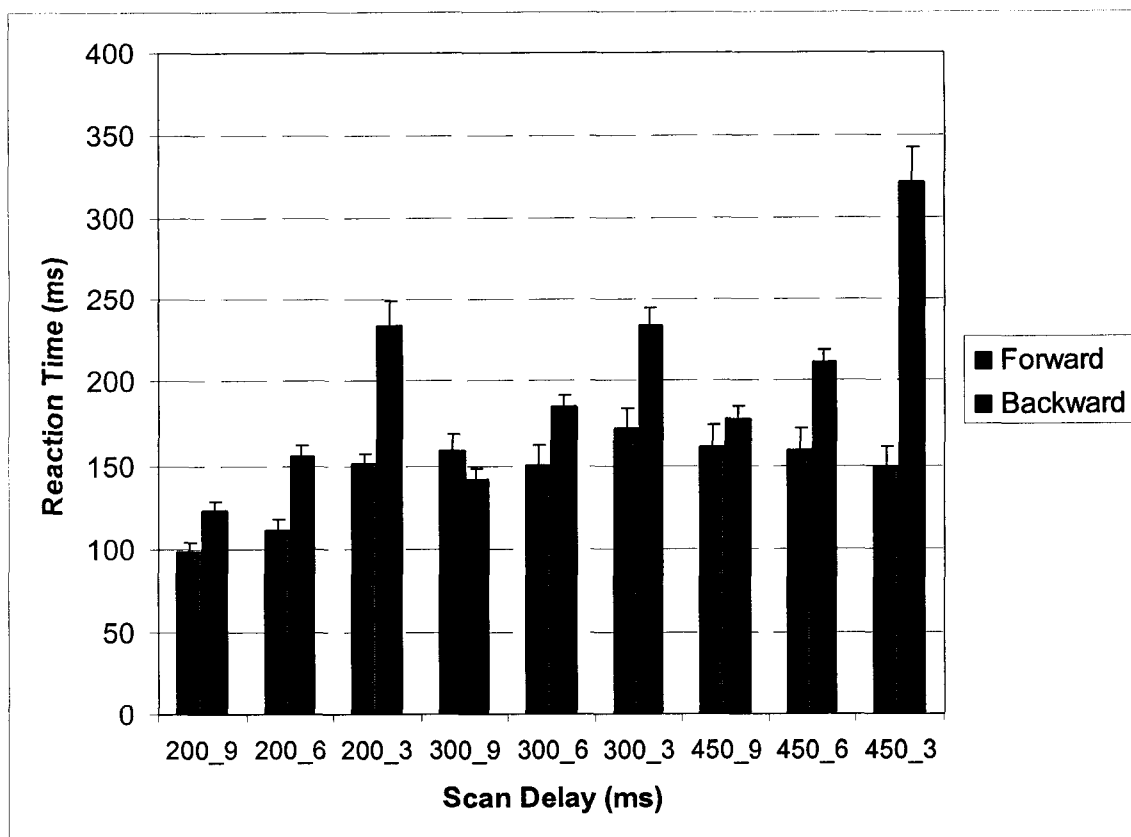


Figure 4.20 Reaction Time vs. Scan Delay for Overscan Tests of Alphabetic Scan

The mean number of errors per test for the backward scan delay of 450 ms was 7.2 ($s = 4.0$) for the forward speed divisor of three, 8.3 ($s = 3.1$) for the forward speed divisor of six, and 11.6 ($s = 5.8$) for the forward speed divisor of nine. The mean number of errors per test for the backward scan delay of 300 ms was 6.0 ($s = 3.2$) for the forward speed divisor of three, 13.0 ($s = 6.1$) for the forward speed divisor of six, and 13.6 ($s = 5.4$) for the forward speed divisor of nine. The mean number of errors per test for the backward scan delay of 200 ms was 22.0 ($s = 10.0$) for the forward speed divisor of three, 18.3 ($s = 9.4$) for the forward speed divisor of six, and 16.1 ($s = 6.4$) for the forward speed divisor of

nine. The mean number of errors per test is displayed in Figure 4.21. Note that for backward scan delays of 450 ms and 300 ms, the errors per test increase as the forward speed divisor increases. However for the backward scan delay of 200 ms, the mean error rate decreases as the scan delay increases.

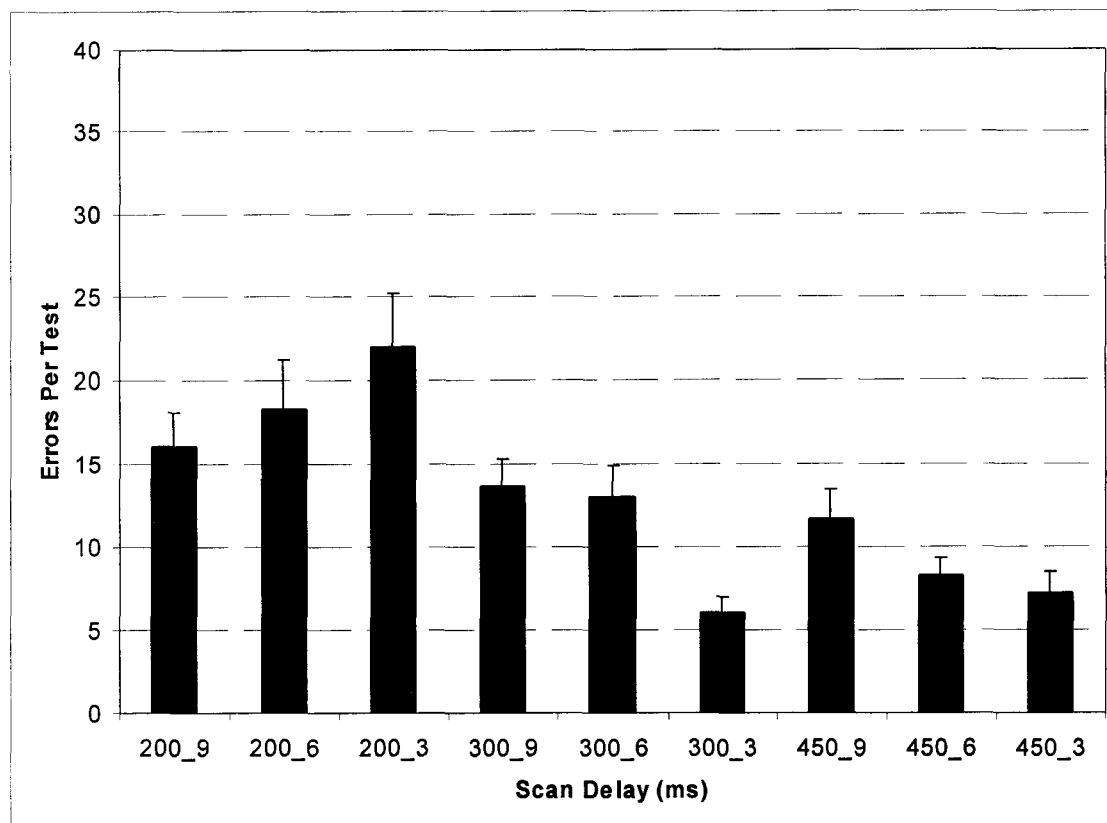


Figure 4.21 Errors vs. Scan Delay for All Overscan Tests of Alphabetic Scan

4.2.4 All Alphabetic Interface Tests

The completion times for overscan and row-column scanning were similar, while users took much longer to complete the scan using the linear technique. The mean completion time for the linear technique was 250.4 s ($s = 58.1$), the completion time for the row-column technique was 143.5 s ($s = 24.4$),

and the completion time for the overscan technique was 155.6 s ($s = 27.9$).

Completion times for all three scanning techniques are displayed in Figure 4.22.

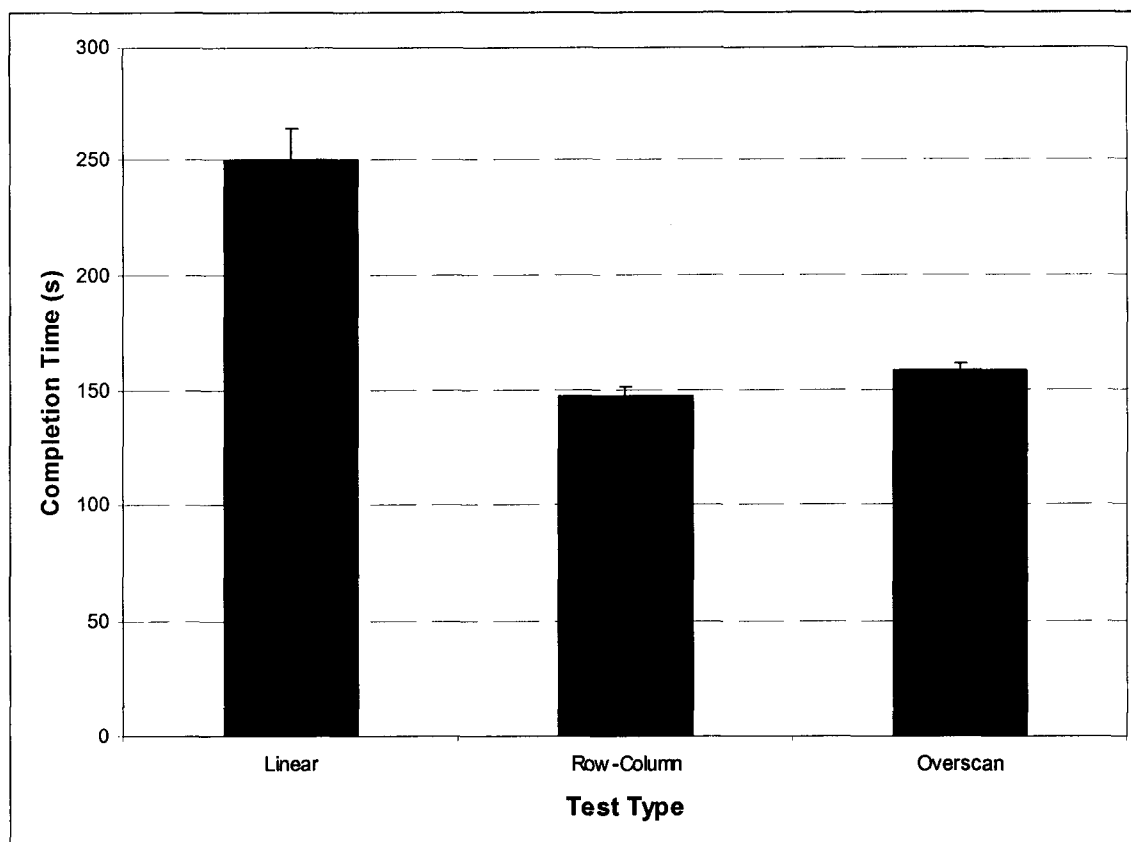


Figure 4.22 Completion Time for All Tests of Alphabetic Scan

Figure 4.23 shows the mean forces measured for all tests of the alphabetic scan. Participants used the least amount of force to operate the switch when using the linear scanning technique, but the force did not vary greatly between the three interfaces. The mean force for the linear technique was 3.0 N ($s = 0.7$), the force for the row-column technique was 3.3 N ($s = 1.0$), and the force for the overscan technique was 3.6 N ($s = 1.0$).

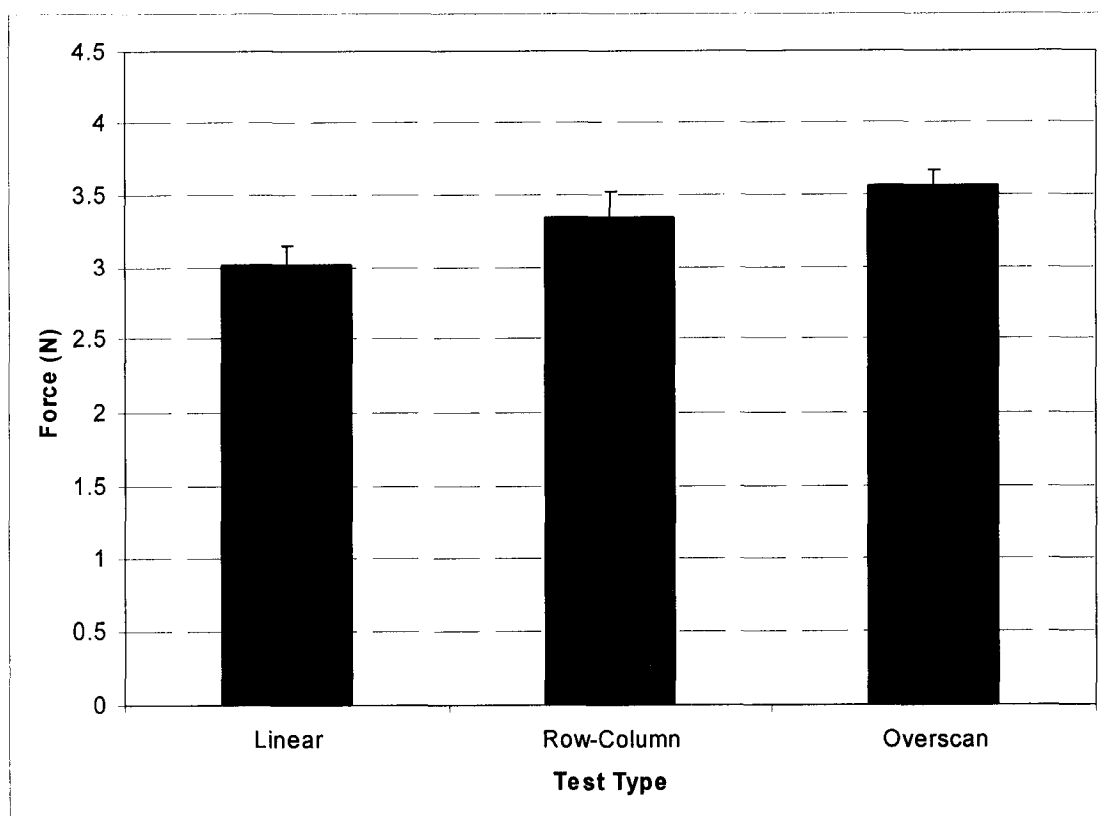


Figure 4.23 Mean Force for All Tests of Alphabetic Scan

The mean key press time for each of the three tests was very similar. Users did not press the activation switch for a longer or shorter duration with any of the scanning techniques. The mean key press time for the linear technique was 198.3 s ($s = 127.8$), the key press time for the row-column technique was 197.2 s ($s = 116.4$), and the key press time for the overscan technique was 210.6 s ($s = 119.7$). Figure 4.24 shows the relationship between key press time and scan type.

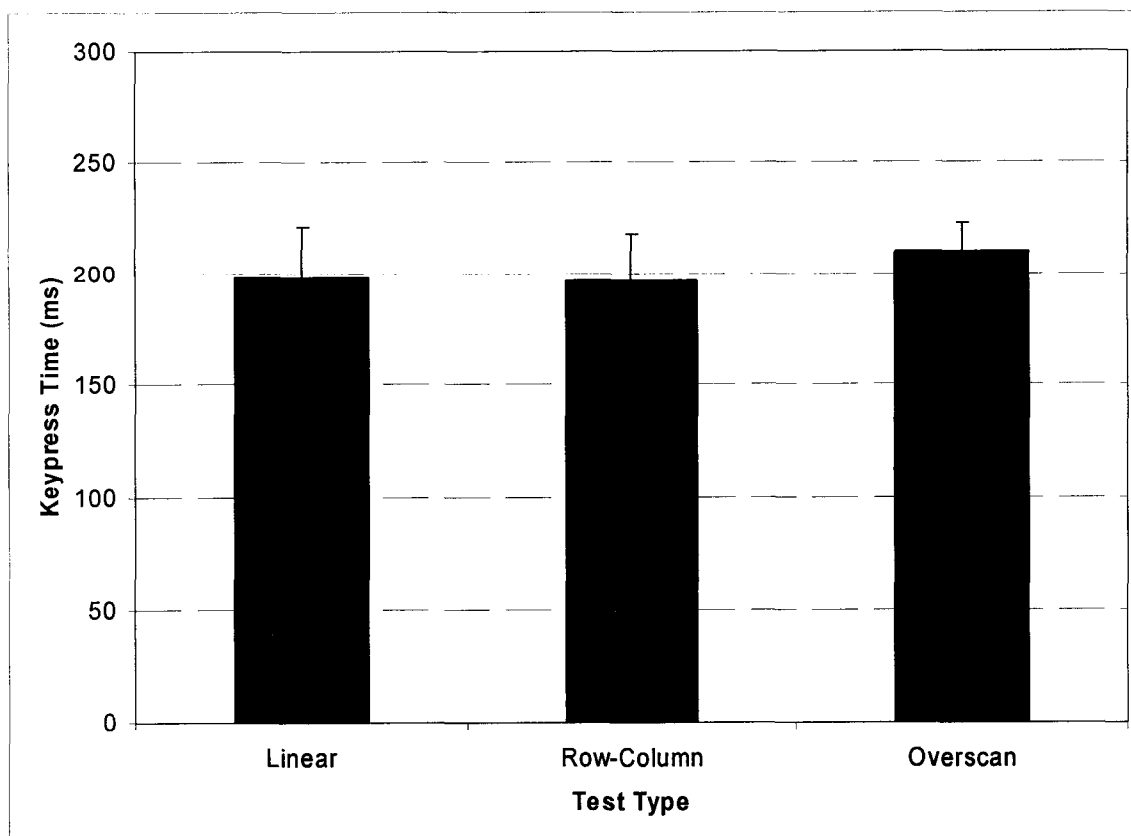


Figure 4.24 Key Press Time for All Tests of Alphabetic Scan

The mean reaction times for the linear and row-column tests were very similar, but the mean reaction time for the overscan test was greater than either of the other tests. This greater reaction time for the overscan technique is likely caused by the changing scan delays of the forward and backward scans which prevent the user from becoming accustomed to one speed. The reaction time for the linear technique was 147.2 ms ($s = 42.8$), the reaction time for the row-column technique was 148.6 ms ($s = 42.7$), and the reaction time for the overscan technique was 172.1 ms ($s = 42.5$).

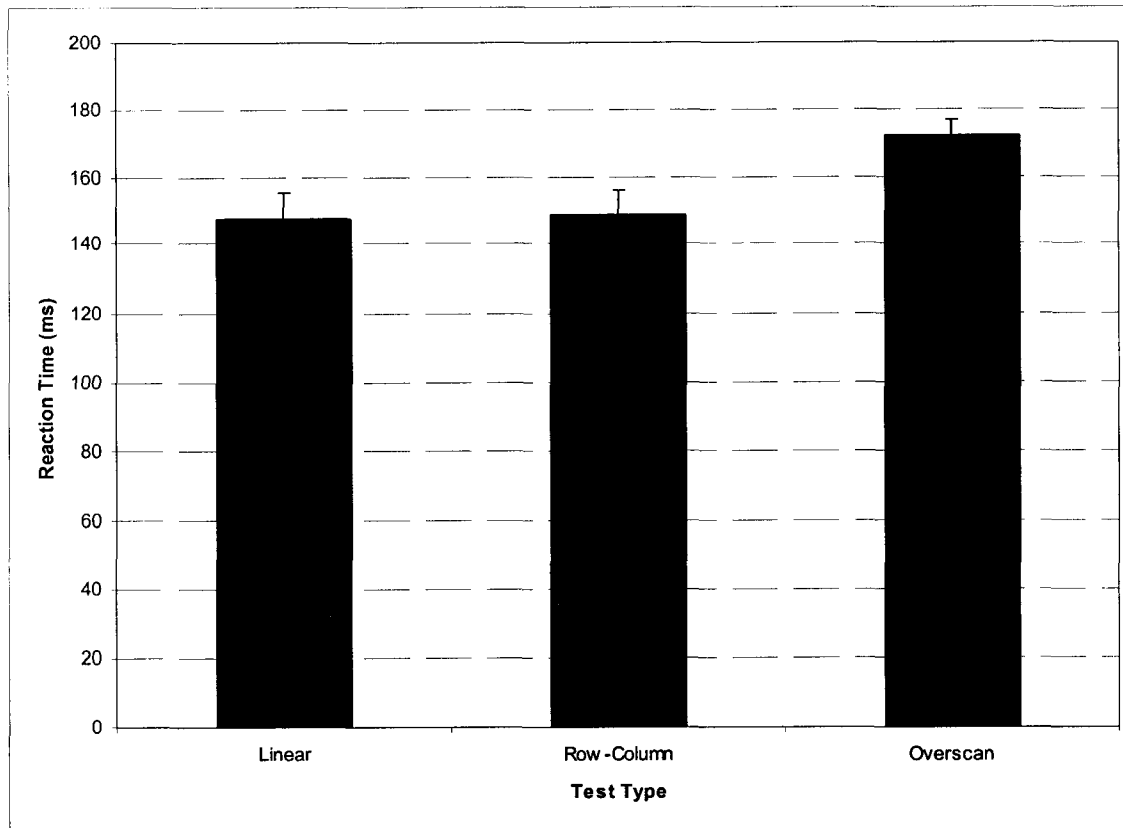


Figure 4.25 Reaction Time for All Tests of Alphabetic Scan

As expected, users had the lowest error rates while using the linear scanning technique, followed by the overscan technique, and then row-column scanning. The mean number of errors per test for the linear technique was 4.9 ($s = 3.9$), the mean number of errors per test for the row-column technique was 16.1 ($s = 14.0$), and the mean number of errors per test for the overscan technique was 12.9 ($s = 7.9$). Error rates for all tests of the alphabetic scan are shown in Figure 4.26.

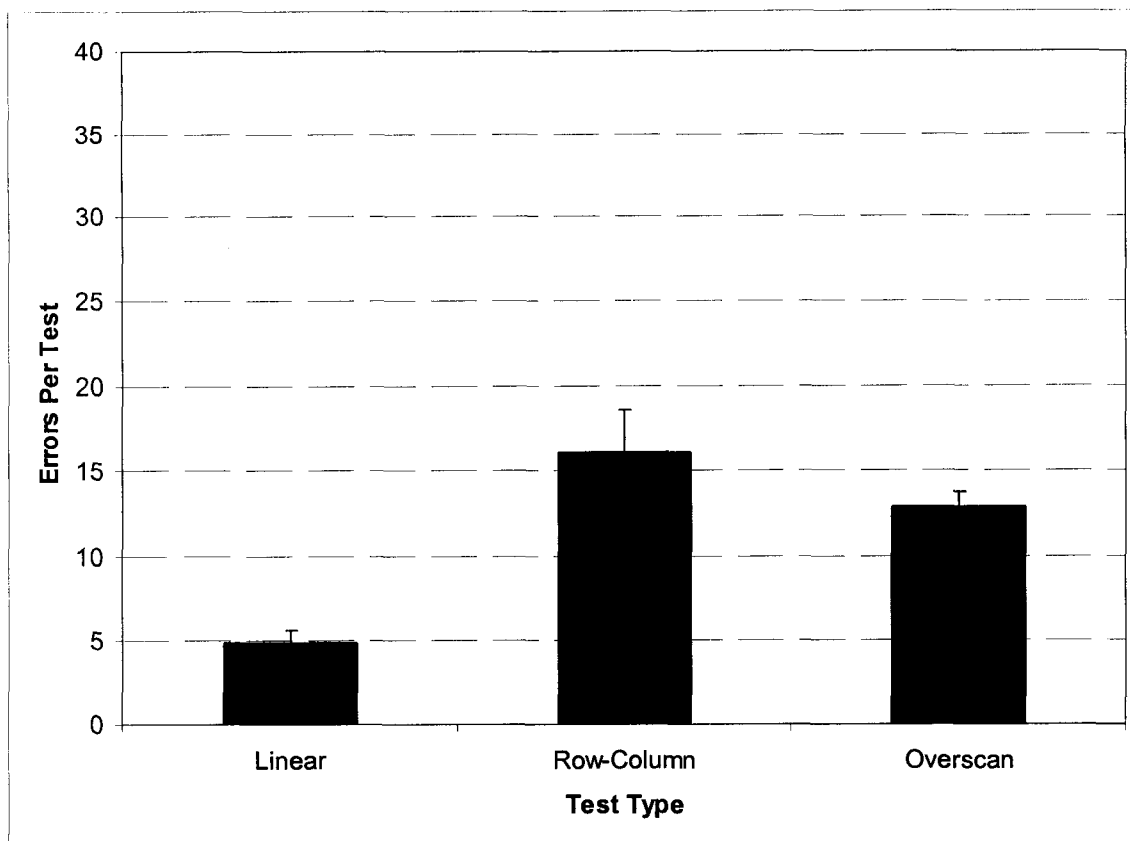


Figure 4.26 Errors for All Tests of Alphabetic Scan

4.3 Internet Browser

4.3.1 Linear Internet Interface

The relationship between completion time and scan delay for the linear tests of the alphabetic scan is displayed in Figure 4.27. Users completed the task in an average time of 67.0 s ($s = 7.1$) at a scan delay of 450 ms. The average completion time dropped to 52.0 s ($s = 7.4$) for a scan delay of 300 ms, and 41.6 s ($s = 7.4$) for a scan delay of 200 ms.

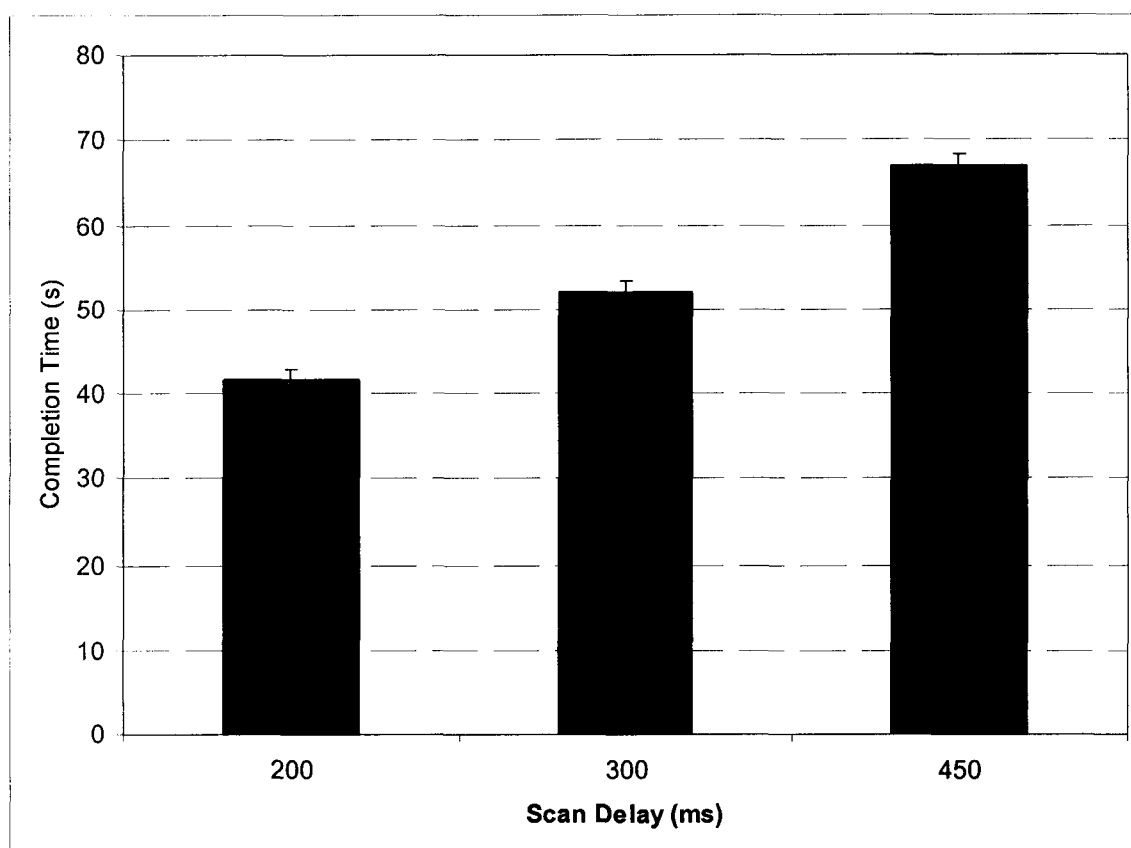


Figure 4.27 Completion Time vs. Scan Delay for All Linear Tests of Internet Scan

In the same fashion as the previous tests, the mean force did not change for different scan delays. Figure 4.28 shows that the mean force stayed relatively constant for all Internet browsing tests utilizing the linear scanning technique. The mean force at 450 ms was 2.9 (s =1.4), the force at 300 ms was 2.8 (s =1.1), and the force at 200 ms was 2.8 (s =1.1).

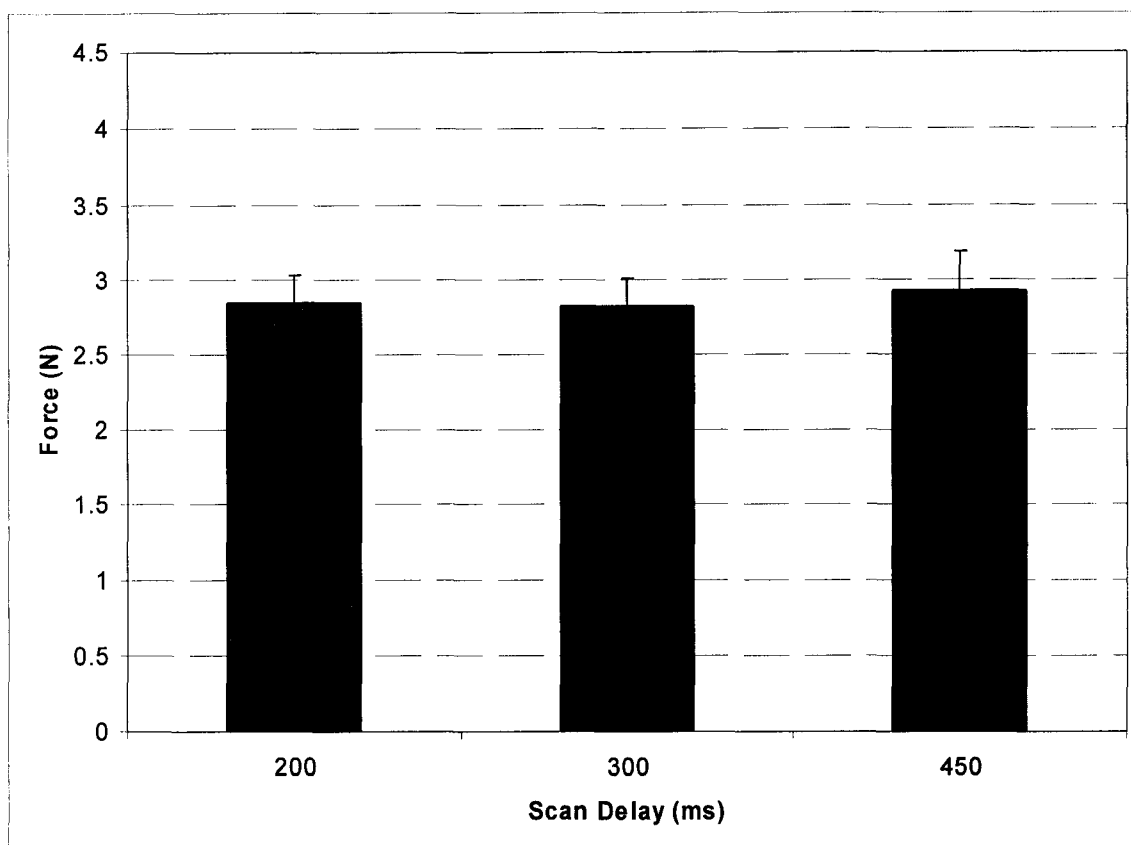


Figure 4.28 Force vs. Scan Delay for All Linear Tests of Internet Scan

The key press time did not change significantly as the scan delay changed, and all means fall well within the standard errors of each other. For a scan delay of 450 ms the mean key press time was measured at 156.8 ms (s = 66.5), at 300 ms the mean key press time was 147.0 ms (s = 67.6), and at 200 ms the mean key

press time was 143.8 ms ($s = 66.9$). The relationship of key press time to scan delay is shown in Figure 4.29.

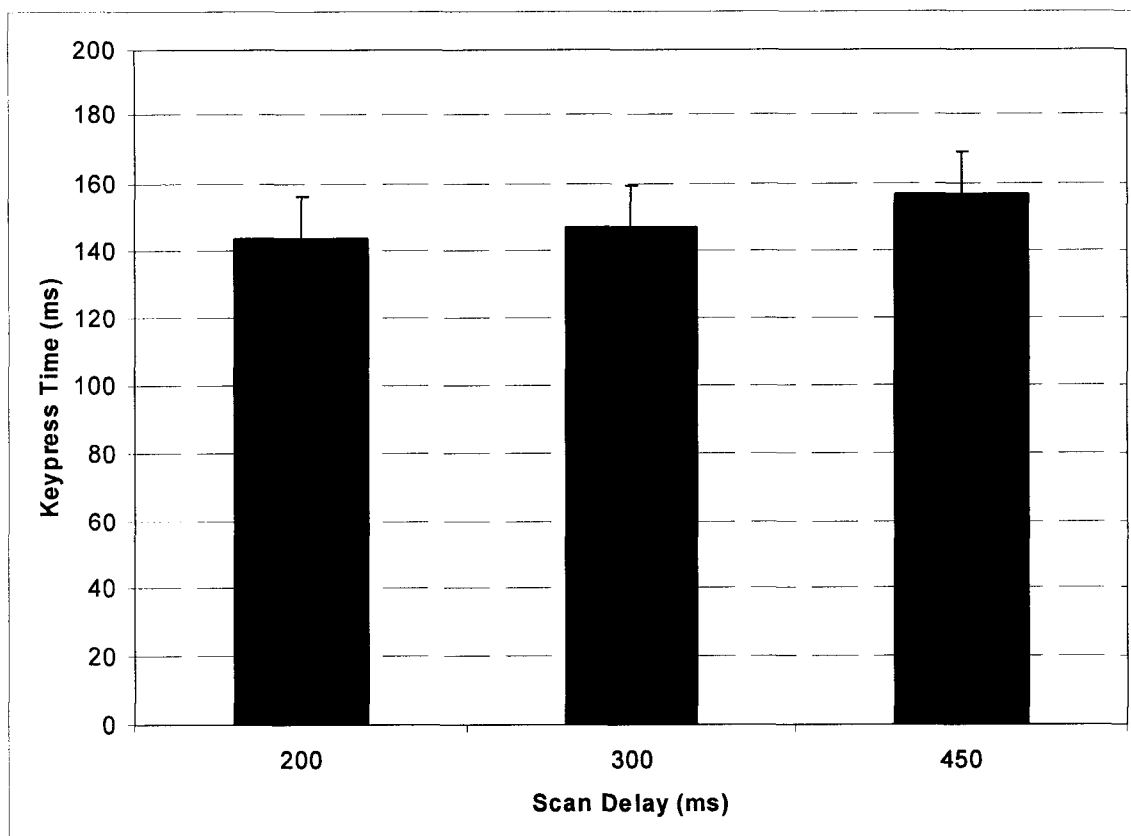


Figure 4.29 Key Press Time vs. Scan Delay for All Linear Tests of Internet Scan

The mean reaction time at a scan delay of 450 ms was 215.2 ms ($s = 43.4$), the reaction time at a scan delay of 300 ms was 186.5 ms ($s = 41.2$), and the reaction time at a scan delay of 200 ms was 111.5 ms ($s = 28.7$). For the linear Internet interface, the ratio of reaction time to scan delay did not follow the same pattern as it did for the linear alphabetic interface. The ratio did increase from a mean reaction time of 450 ms to 300 ms, but then the ratio decreased again when

the scan delay drops to 200 ms. The mean reaction time, shown in Figure 4.30, decreased with decreasing scan delays.

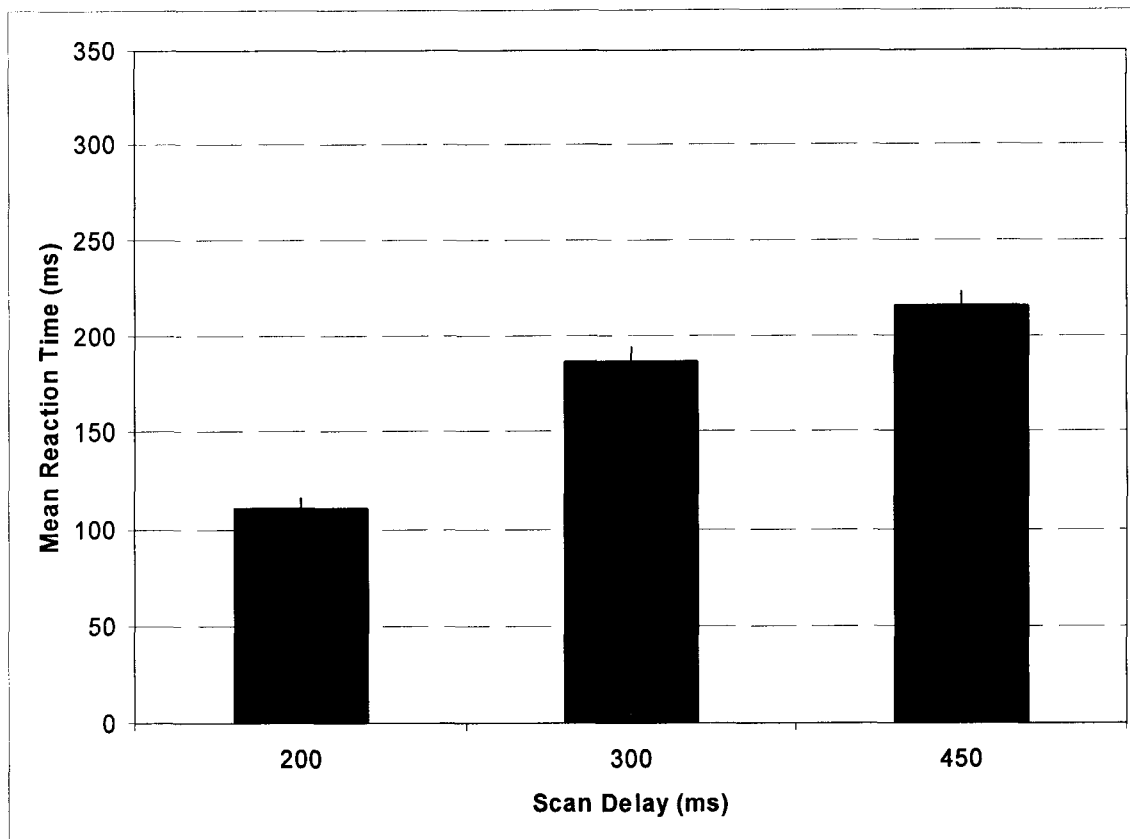


Figure 4.30 Reaction Time vs. Scan Delay for All Linear Tests of Internet Scan

The error rates for the linear Internet scanning tests were similar for scan delays of 450 ms and 300 ms. This pattern is similar to the pattern for error rates of the linear alphabetic scan. The error rate for 450 ms was 0.3 ($s = 0.8$) errors per test, and the error rate for a scan delay of 300 ms was 0.4 ($s = 0.7$) errors per test. However, error rates at a scan delay of 200 ms were more than twice the error

rates for the tests at the higher scan delays. The error rate at a scan delay of 200 ms was 1.1 ($s = 1.2$) errors per test, as shown in Figure 4.31.

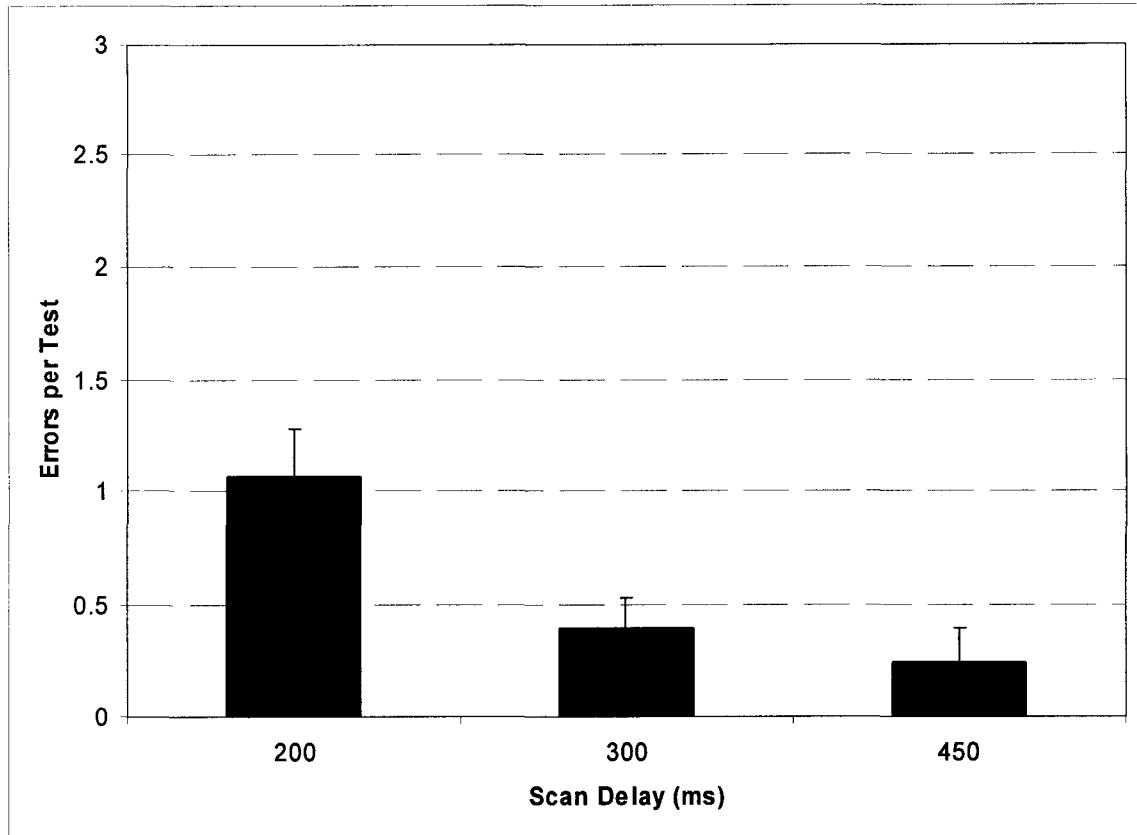


Figure 4.31 Errors vs. Scan Delay for All Linear Tests of Internet Scan

4.3.2 Overscan Internet Interface

Figure 4.32 shows the completion times for all scan delays of the Internet overscan tests. In a pattern similar to the alphabetic overscan completion times, the completion times for the Internet overscan test with a forward speed three times faster than the backward speed were significantly different than the completion times for scan delays with a forward speed six or nine times faster than the backward speed. Note that the completion times actually decreased when the forward scan delay goes from six to nine for the backward scan delays of 300 ms and 200 ms. The mean completion time for the backward scan delay of 450 ms was 40.0 s ($s = 6.6$) for the forward speed divisor of three, 31.1 s ($s = 4.3$) for the forward speed divisor of six, and 30.4 s ($s = 4.5$) for the forward speed divisor of nine. The mean completion time for the backward scan delay of 300 ms was 32.0 s ($s = 6.0$) for the forward speed divisor of three, 27.8 s ($s = 3.6$) for the forward speed divisor of six, and 29.0 s ($s = 7.7$) for the forward speed divisor of nine. The mean completion time for the backward scan delay of 200 ms was 34.5 s ($s = 9.2$) for the forward speed divisor of three, 29.1 s ($s = 7.5$) for the forward speed divisor of six, and 29.9 s ($s = 8.9$) for the forward speed divisor of nine.

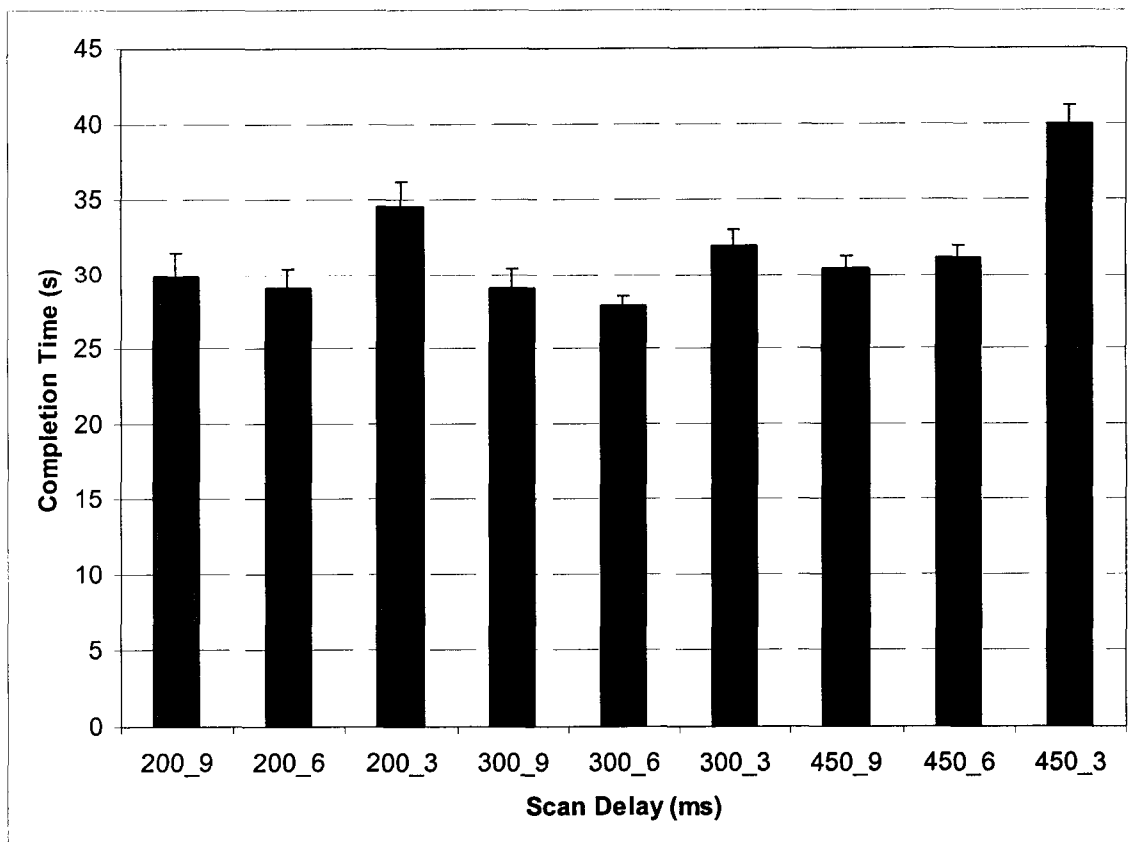


Figure 4.32 Completion Time vs. Scan Delay for All Overscan Tests

The mean forces for the overscan interface did not follow a well defined pattern. The mean force for the backward scan delay of 450 ms was 3.0 N ($s = 1.2$) for the forward speed divisor of three, 3.3 N ($s = 1.2$) for the forward speed divisor of six, and 3.5 N ($s = 1.4$) for the forward speed divisor of nine. The mean force for the backward scan delay of 300 ms was 3.3 N ($s = 1.3$) for the forward speed divisor of three, 3.3 N ($s = 1.2$) for the forward speed divisor of six, and 3.6 N ($s = 1.1$) for the forward speed divisor of nine. The mean force for the backward scan delay of 200 ms was 3.8 N ($s = 1.3$) for the forward speed divisor of three, 3.7 N ($s = 0.9$) for the forward speed divisor of six, and 4.0 N ($s = 1.5$) for

the forward speed divisor of nine. The mean forces measured for the Internet overscan test are shown in Figure 4.33.

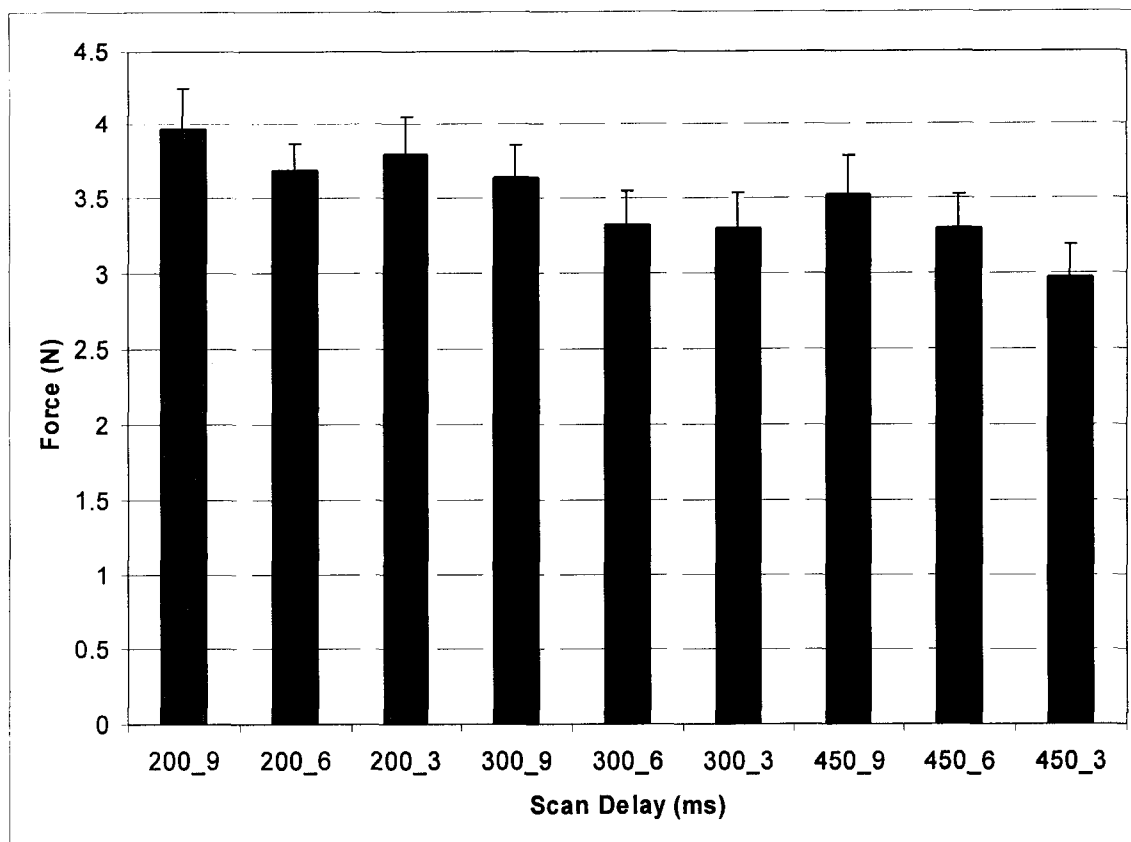


Figure 4.33 Force vs. Scan Delay for All Overscan Tests

Key press times for the Internet overscan tests did not differ statistically for the nine different scan delays. The mean key press time for the backward scan delay of 450 ms was 156.4 ms ($s = 64.8$) for the forward speed divisor of three, 163.9 ms ($s = 63.5$) for the forward speed divisor of six, and 167.8 ms ($s = 59.5$) for the forward speed divisor of nine. The mean key press time for the backward scan delay of 300 ms was 169.7 ms ($s = 67.4$) for the forward speed divisor of

three, 161.0 ms ($s = 70.8$) for the forward speed divisor of six, and 171.6 ms ($s = 68.1$) for the forward speed divisor of nine. The mean key press time for the backward scan delay of 200 ms was 172.1 ms ($s = 70.2$) for the forward speed divisor of three, 167.1 ms ($s = 72.7$) for the forward speed divisor of six, and 169.5 ms ($s = 65.5$) for the forward speed divisor of nine. The standard errors of the means for all mean key press times of the alphabetic overscan tests overlap, as shown in Figure 4.34.

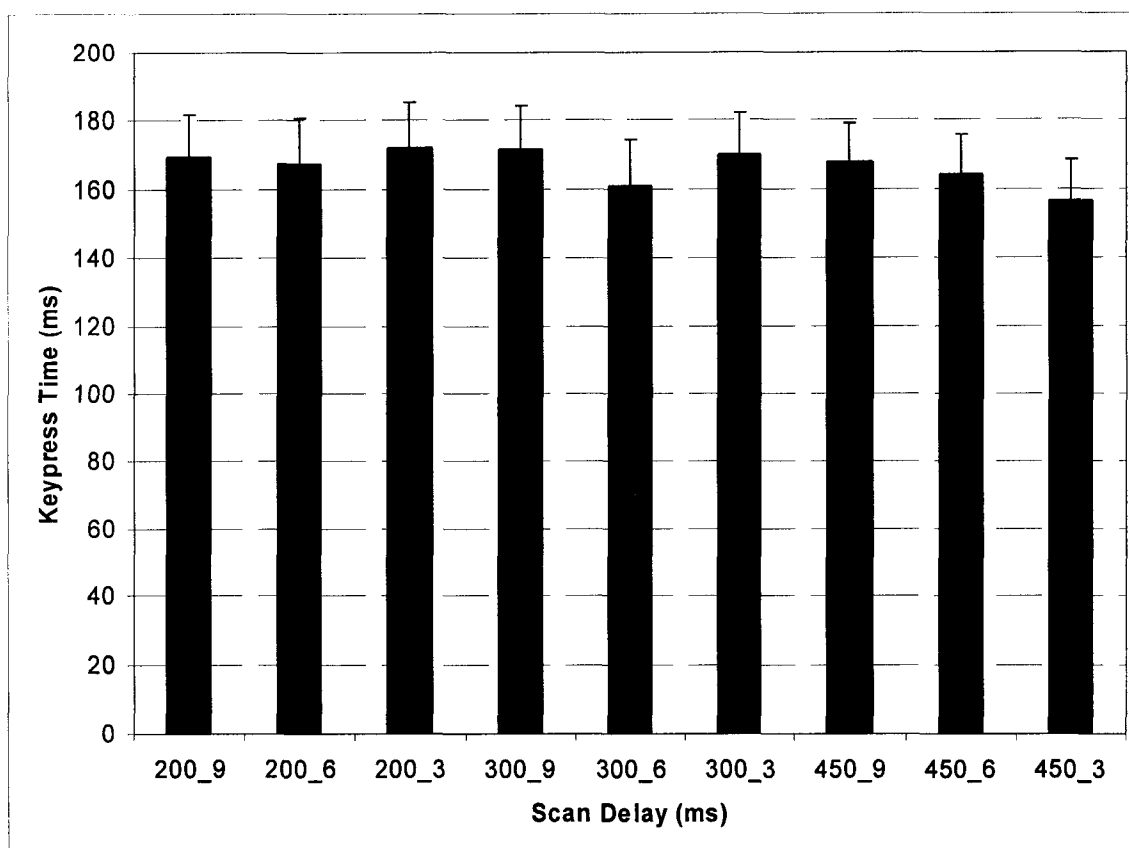


Figure 4.34 Key Press Time vs. Scan Delay for All Overscan Tests

Figure 4.35 shows the relationship between the forward and backward scan delays and the reaction times measured for the alphabetic overscan tests. Reaction times for the backward scan delay decreased as the forward scan delay was decreased for a given backward scan delay. The mean reaction time for the backward scan delay of 450 ms was 370.5 ms ($s = 150.1$) for the forward scan and 233.6 ms ($s = 45.7$) for the backward scan for the forward speed divisor of three, 221.4 ms ($s = 85.7$) for the forward scan and 216.5 ms ($s = 50.5$) for the backward scan for the forward speed divisor of six, and 173.8 ms ($s = 74.3$) for the forward scan and 197.6 ms ($s = 59.9$) for the backward scan for the forward speed divisor of nine. The mean reaction time for the backward scan delay of 300 ms was 282.9 ms ($s = 141.8$) for the forward scan and 88.1 ms ($s = 39.0$) for the backward scan for the forward speed divisor of three, 189.8 ms ($s = 63.3$) for the forward scan and 174.3 ms ($s = 38.1$) for the backward scan for the forward speed divisor of six, and 153.7 ms ($s = 59.0$) for the forward scan and 172.7 ms ($s = 45.0$) for the backward scan for the forward speed divisor of nine. The mean reaction time for the backward scan delay of 200 ms was 244.1 ms ($s = 97.5$) for the forward scan and 131.4 ms ($s = 25.7$) for the backward scan for the forward speed divisor of three, 158.0 ms ($s = 55.9$) for the forward scan and 108.6 ms ($s = 24.9$) for the backward scan for the forward speed divisor of six, and 124.3 ms ($s = 26.7$) for the forward scan and 103.2 ms ($s = 26.9$) for the backward scan for the forward speed divisor of nine.

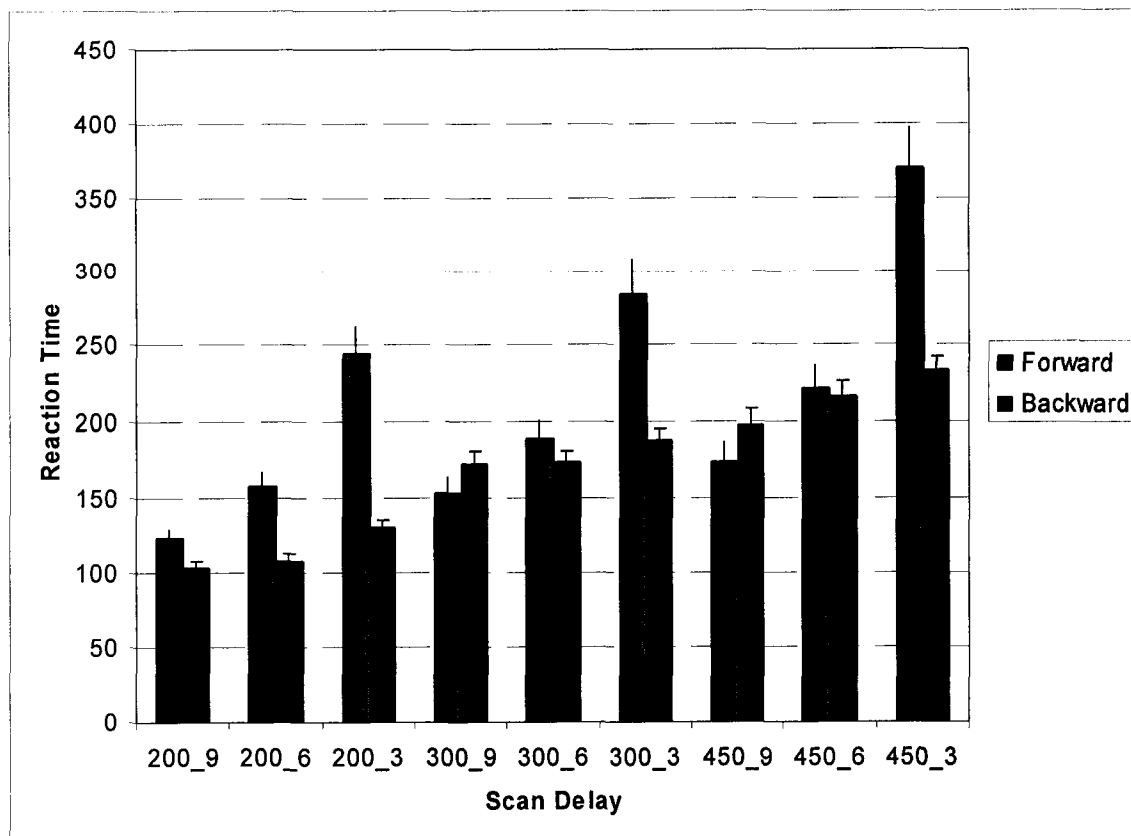


Figure 4.35 Reaction Time vs. Scan Delay for All Overscan Tests

The mean number of errors per test for the backward scan delay of 450 ms was 0.4 ($s = 0.7$) for the forward speed divisor of three, 0.3 ($s = 0.6$) for the forward speed divisor of six, and 0.5 ($s = 0.8$) for the forward speed divisor of nine. The mean number of errors per test for the backward scan delay of 300 ms was 0.6 ($s = 1.0$) for the forward speed divisor of three, 0.6 ($s = 0.7$) for the forward speed divisor of six, and 0.7 ($s = 0.9$) for the forward speed divisor of nine. The mean number of errors per test for the backward scan delay of 200 ms was 2.4 ($s = 2.0$) for the forward speed divisor of three, 1.6 ($s = 1.6$) for the forward speed divisor of six, and 1.5 ($s = 2.1$) for the forward speed divisor of

nine. The mean number of errors per test for the overscan Internet interface is displayed in Figure 4.36. Note that for the backward scan delay of 200 ms, the mean error rate decreases as the forward scan delay increases.

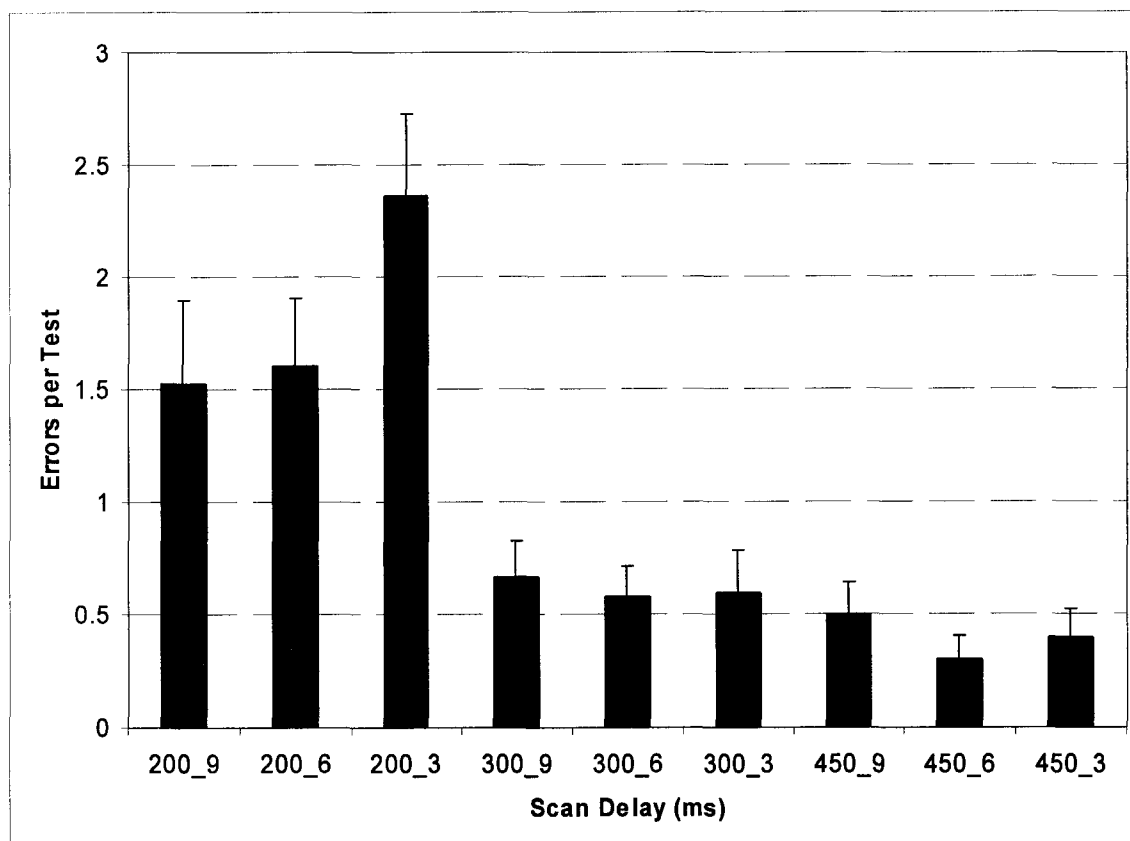


Figure 4.36 Errors vs. Scan Delay for All Overscan Tests

4.3.3 All Internet Browser Interface Tests

The mean completion time for the overscan technique was significantly shorter than the mean completion time of the linear technique. The mean completion time for the linear technique was 53.5 s ($s = 12.7$) and the completion

time for the overscan technique was 31.5 s ($s = 7.5$). Completion times for both scanning techniques are displayed in Figure 4.37.

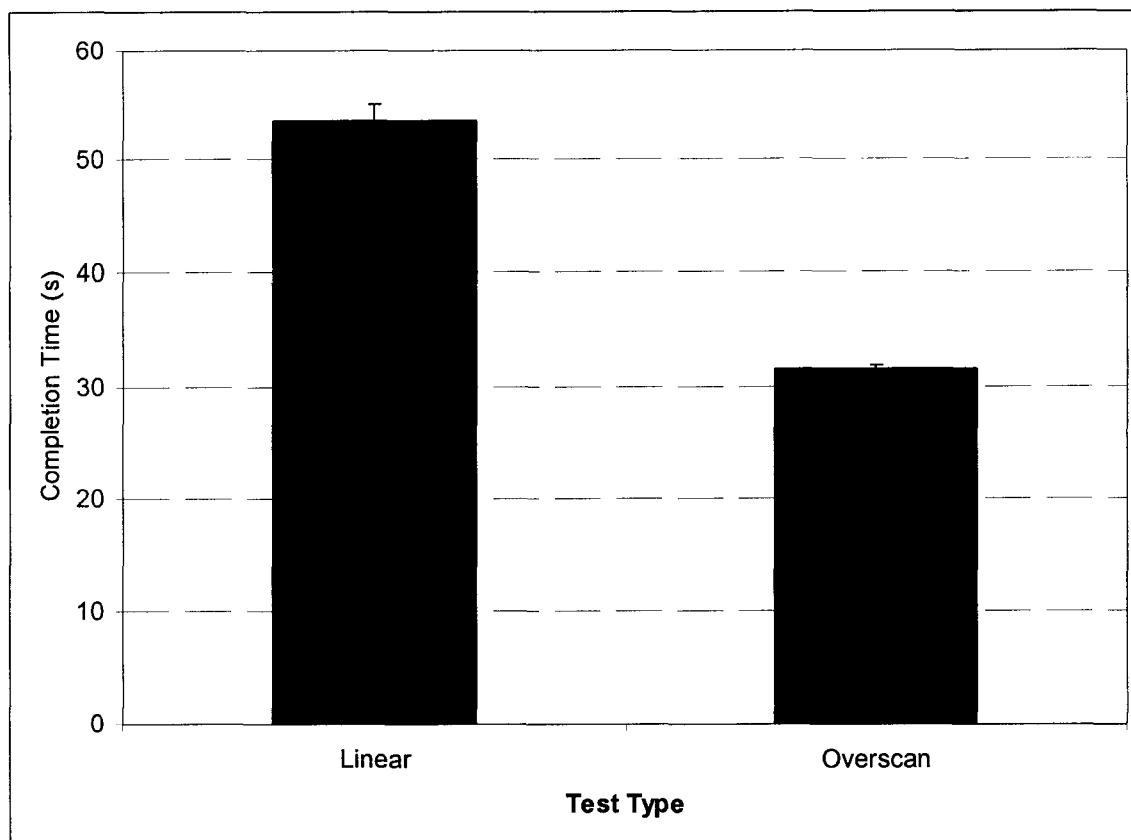


Figure 4.37 Mean Completion Time for All Tests of Internet Scan

The mean force for both techniques tested for the Internet browsing interface is displayed in Figure 4.38. Participants used the least amount of force to operate the switch when using the linear scanning technique. The mean force for the linear technique was 2.7 N ($s = 0.8$) and the mean force for the overscan technique was 3.4 N ($s = 1.0$).

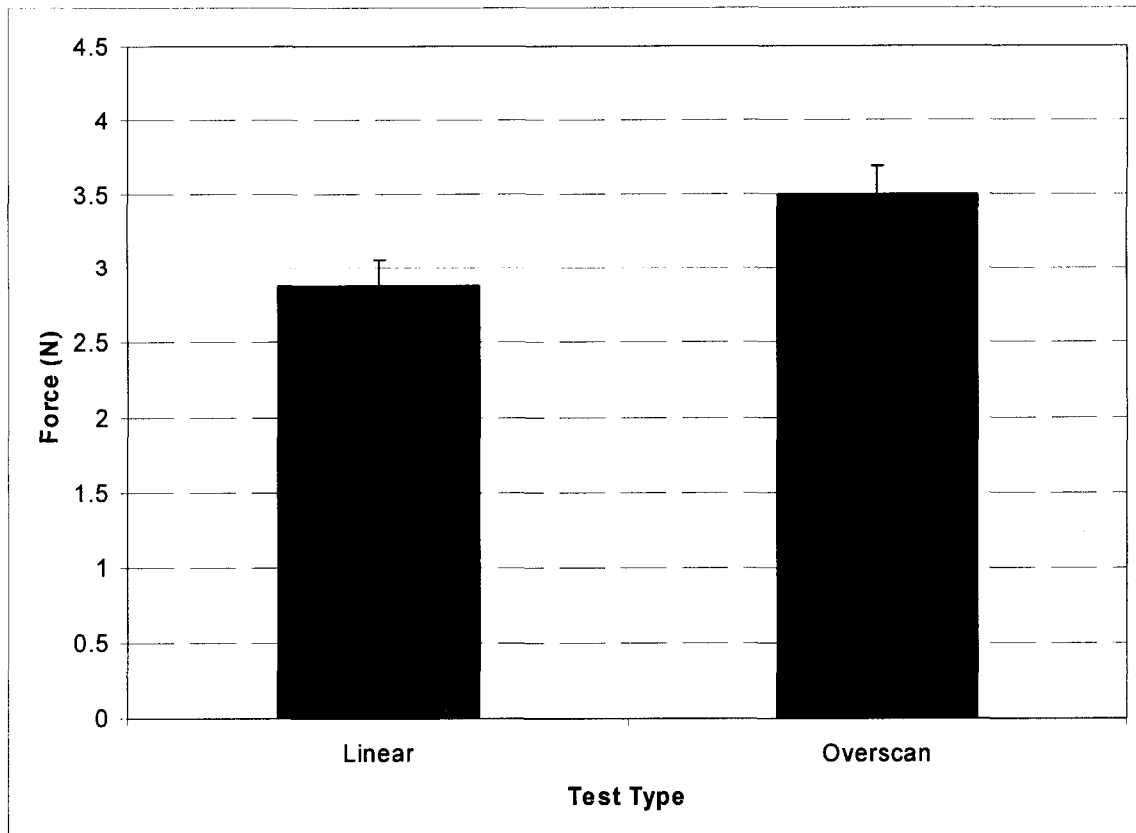


Figure 4.38 Mean Force for All Tests of Internet Scan

The mean key press time for both techniques tested for the Internet browsing interface is displayed in Figure 4.38. Key press times for both tests were similar. The mean key press time for the linear technique was 149.2 s ($s = 61.8$) and the mean key press time for the overscan technique was 166.6 s ($s = 71.0$).

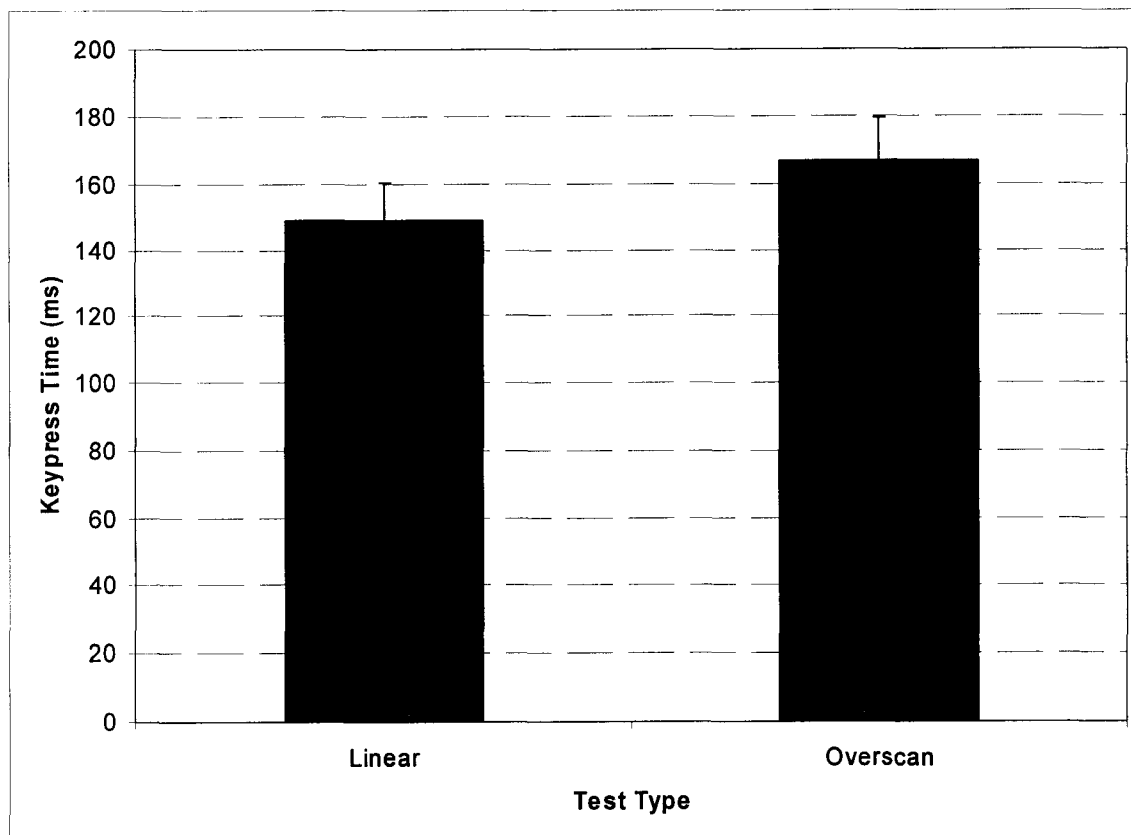


Figure 4.39 Key Press Time for All Tests of Internet Scan

The mean reaction time for users utilizing the overscan technique was greater than the mean reaction time of users utilizing the linear scan technique, as shown in Figure 4.40. This same behavior was observed for the alphabetic scan interface. The mean reaction time for the linear technique was 171.0 ms ($s = 28.5$), and the mean reaction time for the overscan technique was 191.5 ms ($s = 36.7$).

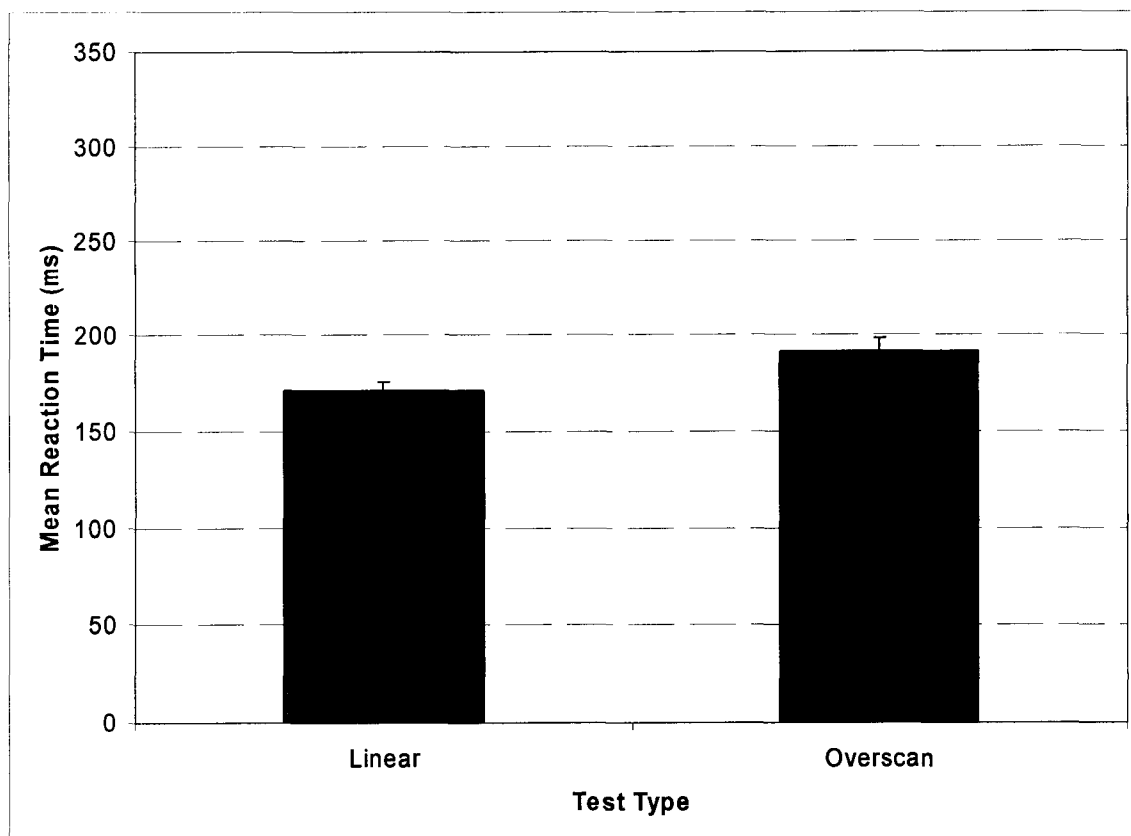


Figure 4.40 Reaction Time for All Tests of Internet Scan

Figure 4.41 shows the mean errors per test for both the linear and overscan techniques of the Internet browser. Just like the alphabetic interface, the Internet interface had a higher error rate for the overscan technique than the linear technique. The mean number of errors per test for the linear technique was 0.5 ($s = 0.5$), and the mean number of errors per test for the overscan technique was 0.9 ($s = 0.5$).

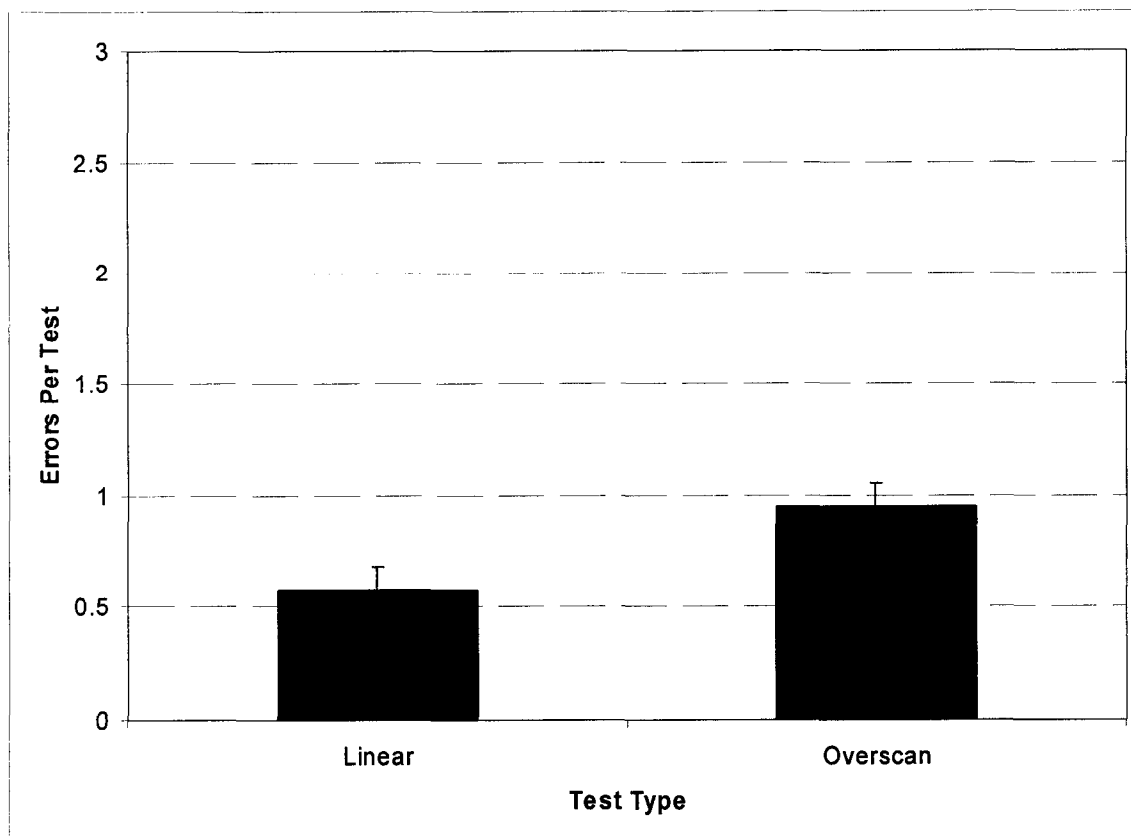


Figure 4.41 Errors for All Tests of Internet Scan

CHAPTER 5

ANALYSIS

5.1 Row-Column Scanning

The data was analyzed using Minitab release 14.20. Data from the initial alphabetic row-column scanning interface was analyzed to determine if there was a *statistical difference* between the forces used during the first 10 switch activations of a test and the last 10 switch activations. Table 5.1 shows that there is a highly statistically significant difference between the means for the two sets of data. This difference is important because it shows that individuals tend to use more force at the beginning of the scan, and use less force once he or she has adapted to the scan.

Table 5.1 Paired T-Test for First 10 Forces vs. Last 10 Forces

Paired T for First10 - Last10				
	N	Mean	StDev	SE Means
First10	61	1.13398	0.28407	0.03637
Last10	61	0.80057	0.24328	0.03115
Difference	61	0.333408	0.31388	0.040188
95% lower bound for mean difference: 0.266267				
T-Test of mean difference = 0 (vs. > 0): T-Value = 8.30 P-Value = 0.000				

Data from the initial experiments was also analyzed to determine how the error rates changed for subjects when the test went from using a scan delay that was "Just About Right" (JAR) to a scan delay that was "Much Too Fast" (MTF). Table 5.2 shows that the error rate increases from a mean of 2.9 errors per test at the JAR speed, to a mean of 27.6 errors per test at the MTF speed. This increase in error rates is expected because as the scan speed decreases, the user has a more difficult time choosing the desired element in the scanning matrix. Because scanning is such a slow input method, any increase in errors makes an already slow process very time-consuming.

Table 5.2 T-Test for Number of Errors Made at JAR and MTF Speeds

Paired T for JAR Errors vs. MTF Errors				
	N	Mean	StDev	SE Mean
JAR Errors	8	2.875	5.0267	1.7772
MTF Errors	8	27.625	19.928	7.0456
Difference	8	-24.75	18.8812	6.6755
95% upper bound for mean difference: -12.1027				
T-Test of mean difference = 0 (vs. < 0): T-Value = -3.71 P-Value = 0.004				

5.2 Linear, Row-Column, and Overscan

For the second set of tests, analysis was again performed using Minitab. For these tests, a repeated-measures ANOVA statistical test was performed to determine the relationship between scan delay and completion time, force, key press time, reaction time, and error rate. The ANOVA output from the Minitab analysis of the completion times for the linear tests is shown in Tables 5.3 through 5.5. These tables show that there is a statistically significant difference in the completion times for the three different scan delays. This decreased completion time is expected because a quicker scan delay allows the user to select the desired element quicker as long as the error rates do not increase to such a level that would negate this time savings.

Table 5.3 First Section of Minitab ANOVA Output

General Linear Model: Completion Time versus Scan Delay, Subject			
Factor	Type	Levels	Values
Scan Delay	fixed	3	200, 300, 450
Subject	random	10	User1 - User10

Table 5.4 Second Section of Minitab ANOVA Output

Analysis of Variance for Completion Time						
Source	DF	Seq SS	Adj SS	Adj MS	F	P
Scan Delay	2	144728	144728	72364	400.96	0.000
Subject	9	2436	2436	271	1.5	0.222
Error	18	3249	3249	180		
Total	29	150413				

Table 5.5 Third Section of Minitab ANOVA Output

Tukey Simultaneous Tests				
Response Variable Completion Time				
All Pairwise Comparisons among Levels of Scan Delay				
Scan Delay = 200 subtracted from:				
	Difference of Means	SE of Difference	T-Value	Adjusted P-Value
Scan Delay				
300	52.5	6.008	8.738	0.000
450	166.4	6.008	27.697	0.000
Scan Delay = 300 subtracted from:				
	Difference of Means	SE of Difference	T-Value	Adjusted P-Value
Scan Delay				
450	113.9	6.008	18.96	0.000

Repeated-measures ANOVA tests were run for all of the data displayed in the Linear, Row-Column, and Overscan Testing Results section of this dissertation. Table 5.6 shows that there is a statistically significant difference in the mean completion times, reaction times, and error rates for each scan delay. However, mean forces and key press times are not significantly different for the different scan delays.

Table 5.6 ANOVA Analysis for Linear Alphabetic Scan

	F	p
Completion Time vs. Scan Delay	400.96	0.000
Force vs. Scan Delay	1.7	0.211
Key press Time vs. Scan Delay	0.64	0.538
Reaction Time vs. Scan Delay	39.75	0.000
Errors vs. Scan Delay	12.79	0.000

ANOVA results for the row-column scan, represented by Table 5.7 shows a similar pattern to the linear scan data. However, the force does vary with scan delay for the row-column tests. Figure 4.13 shows a small increase in force as the scan delay is decreased. It is hypothesized that users had more trouble with the row-column test, as suggested by increased error rates, and therefore pressed the switch with greater force as the difficulty increased.

Table 5.7 ANOVA Analysis for Row-Column Alphabetic Scan

	F	p
Completion Time vs. Scan Delay	4.39	0.028
Force vs. Scan Delay	10.6	0.001
Key press Time vs. Scan Delay	1.42	0.268
Reaction Time vs. Scan Delay	21.32	0.000
Errors vs. Scan Delay	17.63	0.000

ANOVA results for the alphabetic overscan interface, shown in Table 5.8 follow the same pattern as Table 5.7. The mean key press times are not statistically different when users operated the alphabetic overscan interface, and all the other measured parameters were statistically different.

Table 5.8 ANOVA Analysis for Overscan Alphabetic Scan

	F	p
Completion Time vs. Scan Delay	13.1	0.000
Force vs. Scan Delay	3.17	0.004
Key press Time vs. Scan Delay	1.5	0.171
Reaction Time vs. Scan Delay	39.75	0.000
Errors vs. Scan Delay	9.79	0.000

ANOVA results for all tests of the alphabetic scan are shown in Table 5.9.

For this analysis, the data was grouped by scan type. It should be noted that although there is a statistically significant difference between the mean completion times and forces of linear tests compared to the other two tests, the completion times and forces of the overscan and row-column tests were not statistically different. Additionally, the mean reaction times for linear and row-column tests were not statistically different from each other, however the mean reaction time for both of these tests was statistically different from the reaction times of the alphabetic overscan interface.

Table 5.9 ANOVA Analysis for All Scan Tests of Alphabetic Scan

	F	p
Completion Time vs. Scan Delay	71.99	0.000
Force vs. Scan Delay	11.78	0.000
Key press Time vs. Scan Delay	4.51	0.013
Reaction Time vs. Scan Delay	6.57	0.002
Errors vs. Scan Delay	15.06	0.000

5.3 Internet Browser

The data for the Internet browser scan was also analyzed using Minitab. The results for the linear Internet scan, shown in Table 5.10, mirror the findings from the linear alphabetic scan. The mean completion time, mean reaction time, and error rate all vary with the scan delay. Mean key press time and mean force do not vary in a statistically significant manner with the scan delays tested.

Table 5.10 ANOVA Analysis for Linear Scan of Internet Browser

	F	p
Completion Time vs. Scan Delay	112.55	0.000
Force vs. Scan Delay	0.33	0.723
Key Press Time vs. Scan Delay	2.28	0.111
Reaction Time vs. Scan Delay	186.43	0.000
Errors vs. Scan Delay	14.62	0.000

The ANOVA analysis of the overscan Internet experiments is shown in Table 5.11. This data follows the same patterns as the row-column alphabetic scan and the alphabetic overscan experimental analysis. The means of all measured variables are statistically significant for the different scan delays with the exception of key press time.

Table 5.11 ANOVA Analysis for Overscan of Internet Browser

	F	p
Completion Time vs. Scan Delay	13.22	0.000
Force vs. Scan Delay	5.64	0.000
Key Press Time vs. Scan Delay	1.56	0.137
Reaction Time vs. Scan Delay	94.74	0.000
Errors vs. Scan Delay	17.63	0.000

Table 5.12 shows the ANOVA analysis for the comparison of the overscan and linear scanning technique for Internet browsing. This analysis shows that the differences between means for all measured variables are statistically significant. This is the same pattern observed during the analysis of the alphabetic scanning technique. This examination provides statistical validation to the graphical evidence shown in Figure 4.37 through Figure 4.41, which suggests that the means for the measured variables are statistically different.

Table 5.12 ANOVA Analysis for All Scan Tests of Internet Browser

	F	p
Completion Time vs. Scan Delay	443.5	0.000
Force vs. Scan Delay	41.78	0.000
Key Press Time vs. Scan Delay	31.83	0.000
Reaction Time vs. Scan Delay	7.21	0.008
Errors vs. Scan Delay	8.4	0.004

5.4 GOMS Modeling

The GOMS models developed in Chapter 3 were tested using data from one user to verify that the total completion time predicted by the model was similar to the measured total completion time. This validation will ensure that the model is suitable to be used by clinicians to determine the estimated completion time if the variables in the equation are known or approximated. User 10 was chosen at random, and the data from user 10 was used to test the three GOMS equations.

The measurement of the completion time for all three alphabetic tests started whenever the user first pressed the switch to start the scan, so the preparation time variable equals 0.0 s for this calculation. The value for number of characters variable was computed using the sentence "THE QUICK BROWN FOX JUMPS OVER A LAZY DOG." This sentence contains 42 characters. For the 300 ms linear alphabetic test, user 10 committed five errors for an error rate of 0.119 errors per character. The average target position was 16.31, and the mean reaction time for user 10 was 187.5 ms. The system response time for the linear scanning interface was 500 ms. Inputting these values into Equation 2, the predicted completion time for user 10 was 248 seconds, while the measured completion time for user 10 was 244 seconds. The difference between the predicted completion time and the measured completion time is due to the distribution of the erroneous selections made. The erroneous selections did not have the same mean position as the mean target position of the correct

characters, and this difference gave rise to the slight difference between the two times. Still, the model was off by only 1.64%. For a large enough sample size, the mean error positions will likely approximate the mean target position; therefore the clinician should obtain a reasonably accurate prediction for completion times using the model.

For the 300 ms row-column interface, user 10 committed 13 errors for an error rate of 0.31 errors per character. The average row of the target character was 3.17, and the average column of the target character was 3.31. The mean reaction time for user 10 was 188.42 ms. The system response time for the row-column scanning interface was 500 ms. Inputting these values into Equation 3, the predicted completion time for user 10 was 150 seconds, which is different from the measured completion time of 151 seconds by only 0.33%.

Finally, for the overscan interface with a forward scan delay of 50 ms and a backward scan delay of 300 ms, user 10 committed 18 errors for an error rate of 0.43 errors per character. The average target position was 16.31, the mean forward reaction time for user 10 was 19.75 ms. The mean number of positions past the target character of the selected character was 3.45 positions, and the mean backward reaction time was 186.16 ms. Inputting these values into Equation 4, the predicted completion time for user 10 was 173 seconds, which exactly matches the measured completion time of 173 seconds.

These three models could be used by a clinician to determine the most effective scanning technique for a given individual without requiring the

clinician to test all three interfaces with the client. If the clinician had a tool available that was able to measure the error rates and reaction times for an individual, these values could be plugged into the model to get a general idea of which interface would work the best with a specific individual. Such a tool would decrease the amount of time necessary to determine which interface worked best in each individual situation, and this increased time could be spent on training or testing of alternative interfaces.

CHAPTER 6

CONCLUSIONS

Analysis of the overscan technique shows that this scanning technique has desirable attributes which increase communication/selection rates while minimizing error rates. The communication/selection rate for the overscan technique is significantly greater than the rates for linear scanning. Furthermore, the communication/selection rate for the overscan technique is comparable to that of the row-column technique for a grid-type alphabetic scan. While the error rate of the overscan technique is higher than the error rate for linear scanning, the decreased time necessary to select individual links more than makes up for this error rate.

The overscan technique is a viable scanning technique which is particularly well suited for browsing the Internet. By utilizing the overscan technique to scan Internet pages, users can browse at a rate which is much greater than could be obtained using a simple linear scan. This scanning technique allows the user to view the web page in its intended format, while still providing a considerable rate enhancement over other scanning methods which preserve the native format of a web page. The incorporation of this scanning technique into scanning software and devices for persons with a physical

disability would make the Internet browsing experience less sluggish for these individuals while providing an Internet experience that more closely mirrors that of persons without a physical disability.

The three GOMS models give a clinician the ability to enter user, or client, data into the model and then predict which scanning method will have the highest throughput. This ability is important in the clinical setting where clinicians have a limited amount of time to work with individuals, and must identify the best communication or Internet access method as rapidly as possible. By utilizing the models, clinicians can predict throughput with limited data for each client, reducing the need to spend large amounts of time evaluating each scanning interface separately.

This is the first research that has investigated the overscan technique as a method of Internet browsing. Therefore, much work must be done to optimize the overscan interface for Internet browsing. Future work must be done to determine the scanning parameters and find the best relationship between forward scan delays and backward scan delays which produce an optimal user experience. The ratio between the forward scan delay and the backward scan delay should be examined for a wide variety of timings to determine how the ratio changes as an individual approaches the threshold for a reasonable amount of errors. Furthermore, research must be done to examine the effect of scanning Internet pages which are larger than the screen size to determine the most effective method of utilizing the overscan technique in this circumstance.

REFERENCES

- [1] Abascal, J., Gardezabal, L., and Garay, N. (2004) Optimisation of the selection set features for scanning text input. *Proceedings of the 9th International Conference on Computers Helping People with Special Needs (ICCHP 2004)*, Paris, France, 788--795.
- [2] Atwood, M., Gray, W., & John, B. (1996) *Project Ernestine: Analytic and Empirical Methods Applied to a Real-World CHI problem. Human-Computer Interface Design: Success Stories, Emerging Methods and Real-World Context*. San Francisco: Morgan Kaufmann Publishers.
- [3] Bernstein, L. E. (1988) *The Vocally Impaired: Clinical Practice and Research*. Philadelphia: Grune and Stratton.
- [4] Beukelman, D. R., & Mirenda, P. (1988) *Augmentative and Alternative Communication: Management of Severe Communication Disorders in Children and Adults, Second Edition*. Sydney : Brookes Publishing.
- [5] Blatter, B. M., & Bongers, P. M. (2002) "Duration of Computer Use and Mouse Use in Relation to Musculoskeletal Disorders of Neck or Upper Limb." *International Journal of Industrial Ergonomics*, 30(4), 295-306.
- [6] Boase, J., Horrigan, J. B., Wellman, B., & Rainie, L. (2006) "The Strength of Internet Ties: The Internet and Email Aid Users in Maintaining Their Social Networks and Provide Pathways to Help When People Face Big Decisions." Pew Internet and American Life Project. Retrieved July 28, 2006, from http://www.pewinternet.org/pdfs/PIP_Internet_ties.pdf
- [7] Brogmus, G. (1998) "Cumulative Trauma Disorders of the Upper Extremities: How Big a Problem." *American Pain Society*, 8(1).
- [8] Bushnell, M., & Jones, M. (1994) "Get a Grip: Managing RSI at MIT." *Proceedings of the 22nd Annual ACM SIGUCCS Conference on User Services*, 267-270.
- [9] Card, S., Moran, T., Newell, A. (1993) *The Psychology of Human-Computer Interaction*. Hillsdale, New Jersey: Lawrence Erlbaum Associates.
- [10] Clerkin, P., Cronk, S., & Nimmagadda, P. (2005) "GOMS Modeling of Morse Code." *Proceedings of the RESNA 2005 Annual Conference*.

- [11] Cronk, S., & Schubert, R.W. (1987) "Development of a Real-Time Expert System for Automatic Adaptation of Scanning Rates." *Proceedings of the Tenth Annual Conference on Rehabilitation Technology*. 109-111.
- [12] Cronk, S., & Wang, W. (2002) "Investigating Relationships Between User Performance and Scan Delays in Aids that Scan." *Proceedings of the 2002 Annual Conference of the Rehabilitation Engineering and Assistive Technology Society of North America*.
- [13] Cronk, S., Clerkin, P., Aeddy, P., (2007) "Changes in Switch Activation Force during the Use of Aids that Scan." *Proceedings of the 2007 Annual Conference of the Rehabilitation Engineering and Assistive Technology Society of North America*.
- [14] Cronk, S., Clerkin, P., Aeddy, P., & Besio, W. (2005) "Investigating Changes in Physical Effort When Operating Aids that Scan at Inappropriate Scan Delays." *Proceedings of the 2005 Annual Conference of the Rehabilitation Engineering and Assistive Technology Society of North America*.
- [15] Cronk, S., Nimmagadda, P., & Clerkin, P. (2003) "Effects of Display Characteristics on the Ability of a Disabled User to Operate an Aid that Scans." *Proceedings of the 2003 Annual Conference of the Biomedical Engineering Society*.
- [16] Disability Status: 2000 - Census 2000 Brief Retrieved July 28, 2006, from <http://www.census.gov/hhes/www/disability/disabstat2k/table1.html>
- [17] Ferrer, A., Romero, R., & Alcantud, F. (2000) "Analysis of Disabled User Requirements for a Web Browsing Scanning Selection System." *Proceedings of the International Conference on Computers Helping People with Special needs*. 17-21.
- [18] FlexiForce data sheet. Retrieved August 25, 2006, from <http://www.tekscan.com/flexiforce/flexiforce.html>
- [19] Goossens, C. S. (1992) *Utilizing Switch Interfaces with Children who are Severely Physically Challenged*. Austin, TX: Pro-Ed.
- [20] Hayes, M. G. (1998) "Individuals with Disabilities Using the Internet: A Tool for Information and Communication." *Technology and Disability*, 8(3), 153-158.
- [21] Hermsdorf, D., Gappa, H., & Pieper, M. (1998) "Webadapter: A Prototype of a WWW Browser with New Special Needs Adaptations." *Proceedings of ICCHP*, 151-160.

- [22] Horstmann, H., & Levine, S. (1990) "Modeling of User Performance with Computer Access and Augmentative Communication Systems for Handicapped People." *Augmentative & Alternative Communication*, 6(4), 231-241.
- [23] Horstmann, H., & Levine, S. (1992) "Modeling AAC User Performance: Response to Newell, Arnott, and Waller (1992)." *Augmentative & Alternative Communication*, 8(2), 92-97
- [24] Horstmann, H., & Levine, S. (1994) "Modeling the Speed of Text Entry with a Word Prediction Interface." *IEEE Transactions on Rehabilitation Engineering*, 2(3), 177-187.
- [25] Jensen C., Borg V., Finsen L., Hansen, K., Juul-Kristensen, B., & Christensen, H. (1998) "Job Demands, Muscle Activity and Musculoskeletal Symptoms in Relation to Work with the Computer Mouse." *Scandinavian Journal of Work, Environmental Health*, 24(5), 418-424.
- [26] Jensen, C., Finsen, L., Sogaard, K., and Christensen, H. (2002) "Musculoskeletal Symptoms and Duration of Computer and Mouse Use." *International Journal of Industrial Ergonomics*, 30(4-5), 265-275.
- [27] John, B. & Kieras, D. (1996) "The GOMS Family of User Interface Analysis Techniques: Comparison and Contrast." *ACM Transactions on Computer-Human Interaction*, 3(4), 320-351.
- [28] John, B. & Kieras, D. (1996) "Using GOMS for User Interface Design and Evaluation: Which Technique?" *ACM Transactions on Computer-Human Interaction*, 3(4), 287-319.
- [29] John, B. (1995) "Why GOMS?" *ACM Interactions*, 2(4), 80-89.
- [30] Kambeyanda, D., Singer, L., & Cronk, S. (1997) "Potential Problems Associated with the Use of Speech Recognition Products." *Assistive Technology*, 9(2), 95-101.
- [31] Karimullah, A.S. & Sears, A. (2002) "Speech-Based Cursor Control." *Proceedings of SIGCAPH 2002*, 178-185.
- [32] Kaye, H. S. "Computer and Internet Use Among People with Disabilities. Disability Statistics Report (13)." Washington DC: U.S. Department of Education, National Institute on Disability and Rehabilitation Research. 2000.

- [33] Keates, S., Clarkson, P., & Robinson, P. (1998) "Developing a Methodology for the Design of Accessible Interfaces." *Proceedings of the 4th ERCIM Workshop*, 1-15.
- [34] Kryger, A.I. et al. (2003) "Does Computer Use Pose an Occupational Hazard for Forearm Pain; from the NUDATA study." *Occupational and Environmental Medicine*, 60, e14.
- [35] LaPlante, M. & Carlson, D.(1996) "Disability in the United States: Prevalence and Causes, 1992. Disability Statistics Report (7)." Washington, DC: U.S. Department of Education, National Institute on Disability and Rehabilitation Research.
- [36] Lawrence R.C. et al. (1998) "Estimates of the prevalence of arthritis and selected musculoskeletal disorders in the United States." *Arthritis and Rheumatism*, 41, 778-799.
- [37] Lesh, G. , Higginbotham, J., & Moulton, B. (2000) "Techniques for automatically updating scanning delays." *Proceedings of the RESNA 2000 Annual Conference: Technology for the New Millennium*. 88-90.
- [38] Lesh, G., Moulton, B. and Higginbotham, J. (1998) "Techniques for Augmenting Scanning Communication." *Augmentative and Alternative Communication*, 14(2), 81-101.
- [39] Lewand, R. (2000) *Cryptological Mathematics*. Cambridge: Mathematical Association of America.
- [40] Liffick, B. W. (2003) "Assistive Technology in Computer Science." *International Symposium on Information and Communication Technologies*.
- [41] Mankoff J., Dey A., Batra U. & Moore, M. (2002) "Web Accessibility for Low Bandwidth Input." *Proceedings of ASSETS 2002*, 17-24.
- [42] National Instruments data sheet. Retrieved August 25, 2006, from <http://sine.ni.com/nips/cds/view/p/lang/en/nid/11914>
- [43] Newell, A., Arnott, J., & Waller, A. (1992) "Further Comment on the Validity of User-Modeling in AAC." *Augmentative & Alternative Communication*, 8(3), 252-253.
- [44] Newell, A., Arnott, J., & Waller, A. (1992) "On the Validity of User-Modeling in AAC: Comments on Horstmann and Levine (1990)." *Augmentative & Alternative Communication*, 8(2), 89-91.

- [45] Pfeiffer, P., & Heintzelman, M. (1993) "Machines, statute, and people: strategies for promoting RSI awareness in computing curricula. Technical Symposium on Computer Science Education Archive." *Proceedings of the Twenty-Eighth SIGCSE Technical Symposium on Computer Science Education*.
- [46] Pierce, J. R. (1980) *An Introduction to Information Theory: Symbols, Signals and Noise* (2nd Ed.). New York: Dover.
- [47] Sears, A., Feng, J., & Oseitutu, K. (2003) "Hands-Free, Speech-Based Navigation During Dictation: Difficulties, Consequences, and Solutions." *Human-Computer Interaction*, 18, 229-257.
- [48] Shum, S., Jorgensen, A., Aboulafia, A., & Hammond, N. (1994) "Communicating HCI Modelling to Practitioners." *Conference Companion on Human Factors in Computing Systems*, 271-272.
- [49] Simpson, R., Koester, H., & LoPresti, E. (2006) "Evaluation of an Adaptive Row/Column Scanning System." *Technology and Disability*. 18(3), 127-138.
- [50] Steriadis, C.E. & Constantinou, P. (2003) "Designing Human-Computer Interfaces for Quadriplegic People." *ACM Transactions of Computer-Human Interaction*, 10(2), 87-118.
- [51] United States Department of Labor. Bureau of Labor Statistics. December 13, 2005. Retrieved August 25, 2006, from <http://www.bls.gov/news.release/pdf/osh2.pdf>
- [52] Vaughan, T.M. et al. (2003) "Guest Editorial Brain-Computer Interface Technology: A Review of the Second International Meeting." *IEEE Trans Neural Systems and Rehabilitation Engineering*, 11(2), 94-109.
- [53] Venkatagiri, H. S. (1999) "Efficient Keyboard Layouts for Sequential Access in Augmentative and Alternative Communication." *Augmentative and Alternative Communication*, 15(2), 126-134.
- [54] Viera, A. J. (2003) "Management of Carpal Tunnel Syndrome." *American Family Physician*, 68(2), 265-272.
- [55] Wilgis, S. E. (2002) "Treatment Options for Carpal Tunnel Syndrome." *Journal of the American Medical Association*, 288, 1281-1282.