Who Can Use What?

Proposals for the Development of a Recommender System for Selecting the Most Appropriate Text Input Method for a Given User Based on the User's Physical Abilities.

By Carl Sigmond
Department of Computer Science
Haverford College
Haverford, PA

Advisor: John P. Dougherty

Abstract. Those who cannot use computers are severely disenfranchised by society's present and future reliance on digital technology. Persons with disabilities comprise a significant portion of this disenfranchised group. For persons with disabilities, one of the biggest obstacles to computer use is the input of text. This thesis aims to be the basis for the development of a recommender system to aid users with a wide array of physical abilities in selecting the most applicable text input method for them. We present a survey of text input devices both for people with disabilities and for people in the general population. We then propose a framework for the development of a full scale recommender system for selecting the most appropriate text input method for a given user based on that user's physical abilities.

I. Introduction

In these modern times, with the reliance on computers and mobile technologies continuing to rise, we as a general society do not tend to think critically about how we interact with computers. Most computer users type on traditional keyboards and use traditional mice, while reading text and visually processing graphics on traditional monitors. These input and output actions, however, are not possible for all members of our society. Those with limited income may not be able to afford computers or have the time or resources to go to public computer stations [Honye, et. al., 2012]. Those with limited fine and/or gross motor skills may not be able to physically use a standard keyboard or mouse or read from a standard monitor. We, as a society, must be aware of these differences in computer use and must work to create equal opportunities for these marginalized populations.

Our goal is that this thesis may lead to a greater understanding of computer use across users with a wide spectrum of physical abilities. There are many input devices in existence for able bodied users and users with disabilities, and research in both of these areas is ongoing. That being said, if people do not know a technology from which they could benefit exists, it is useless to them. Furthermore, given a list of input methods or a web search of same, if people are not able to discern which input device best meets their particular needs, the list or web search is useless. Additionally, for both populations in question, if the device or input method requires much overhead or setup or is prohibitively costly, it is also useless. This thesis serves as a compilation and analysis of text input systems and proposes a framework for the development of a recommender system that can be implemented to assist in the selection of appropriate input devices for users with a wide array of fine and gross motor abilities.

We begin by precisely defining the terms and techniques we will employ throughout this thesis. Then, we present a survey of relevant research on text input methods. Some of these methods are targeted at the general population, while others were designed for use by people with disabilities. We proceed by comparing select input methods to what we deem as the standard and most widespread input method - the traditional computer keyboard - identifying similarities and differences. We then propose a framework for the development of a recommender system that will be able to suggest a ranked order of input devices based on a user's specific abilities, and we will discuss how such a system may be implemented. We hope that our proposed recommender system could help those with and without disabilities determine which input methods may be most beneficial for them. A summary with ideas for further research concludes this thesis.

II. Definitions of Terms

In order to proceed, we must provide a base of knowledge on which our analysis can sit. We define someone who has a physical disability as a person who has a motor condition that substantially inhibits one or more major life activities. This definition is consistent with that of the Americans with Disabilities Act [Americans with Disabilities Act of 1990]. The cause of this condition could be neurological, possibly stemming from the aging process or a physical accident, or it could be from an amputation or other non-neurological impairment. The disabilities referenced in further sections of this thesis include but are not limited to cerebral palsy, Parkinson's Disease, Multiple Sclerosis, and spinal cord injuries that restrict motor control from the neck down. For the purpose of this thesis, we are only concerned with physical disabilities which alter a person's use of a traditional keyboard.

We will be investigating various input methods that may or may not have been designed for people with disabilities but nonetheless are used by this population. Some of these technologies are common knowledge to most computer users and people in academia, while others are more obscure. What follows are definitions of the technologies discussed throughout this thesis:

- *Voice recognition:* automatic interpretation of vocalized words or sounds into computer text or commands
- *Sign Language recognition*: automatic interpretation of standard sign language via cameras and/or sensitized gloves
- *Eye gaze tracking:* use of infrared or other technologies to detect where a user is looking on the screen, and thereby control a mouse, which could interact with an on-screen keyboard
- *Stylus input:* detection of the movement or taps of a stylus across a sensitized surface and interpretation of those movements or taps
- *Handwriting recognition:* automatic interpretation of stylus gestures aimed to simulate paper handwriting
- *Foot movement recognition:* interpretation of foot movement to input text or control cursor or pointer positioning
- *Switch input:* use of a single or limited number of switches to interact with software to input text and/or control user interfaces

Temporal switch input is a form of switch input which relies on the user being able to, not only accurately hit one or more switches, but hit them at particular points in time. The element of time provides for increased functionality with a limited number of switches. Temporal switch input, as well as the other input methods described above will be referenced throughout this thesis.

III. The Survey: Prior and Ongoing Research

The research area for efficient, effective, and easy to use text input is vast and has been active for decades. Today, it is active, spanning a large spectrum from able bodied users, mobile applications, and input methods for people with limited mobility. It is impossible to capture the entire research area in one section of this thesis, but this section contains a representative sampling of different projects that, in one way or another, try to speed up input, increase accuracy and/or enable those who have difficulty using traditional keyboards to be able to input text.

We will discuss input methods that span this wide spectrum. Some projects were specifically designed to benefit people with disabilities while others were designed to benefit the general population. Many in the latter category were designed to be used in mobile computing contexts. These include the Canesta Projection Keyboard and two types of stylus-based input. The projects designed specifically for people with disabilities include two types of on-screen keyboards, a physical keyboard that can adjust its settings dynamically based on analysis of user input, eye-gaze, voice and sign language recognition, and several methods of switch input. It should be noted that there are many other forms of text entry, designed for both people with disabilities and for the general population, but this serves as a representative sampling.

We begin with an analysis of how humans input text into a computer. We will identify the traditional QWERTY keyboard as the standard form of text input and the typical computer user as a person who can type on a traditional keyboard at a competitive speed. A standard unmodified keyboard remains the dominant means of text entry today [Joyce, B., Moxley, R., 1990]. This is in large part due to the dominance of typewriters prior to the commercialization of desktop and laptop computers. For someone who has the standard set of abilities and challenges, using a traditional keyboard presents few obstacles. Someone who has never used a computer keyboard or typewriter may have challenges learning the location and function of the keys, but with practice, they can become proficient and learn to touch type if they desire.

It is well known that people who use a standard keyboard for multiple hours a day are at risk of developing muscle weakness or other conditions resulting from extended use of the keyboard. Measures can be implemented to try and decrease the risk of injury. These measures include obtaining wrist and arm supports, positioning the keyboard so that it is at a more ergonomic location, and obtaining a variant to the standard keyboard that is designed to be more ergonomically suitable. These measures help reduce the risk of injury, but they can also be useful for someone who has already had a typing injury. For the latter group, these measures can enable them to type again without having the injury flare up.

Most common operating systems have built in accessibility and keyboard functionality features where a user can control the behavior of the keyboard. These features include key repeat, key bounce, repeat delay, and a feature commonly know as "sticky keys," which allows a user to use the modifier keys without having to hit multiple keys at once [Trewin, 2004]. These features could benefit someone who has somewhat limited fine motor ability but is able to use a standard keyboard anyway, perhaps with a limited number of fingers.

Keyguards

Users who have limited fine motor control in their fingers may benefit from the use of a keyboard/keyguard. A keyguard is a piece of hard plastic that sits directly over the keyboard, covering the entire surface. Above each key, a hole is cut through the hard plastic keyguard. Users type by resting their hand(s) on the keyguard and poking their fingers through the holes to hit desired keys. The keyguard enables someone, who might otherwise not be accurate enough to type on a standard keyboard, to do so with a minimal amount of adaptation. Drawbacks to the keyboard/keyguard are that typing with it is slow, as poking fingers through the holes of the keyguard takes time, and the fingers are prone to blister and cut. The operating system keyboard and accessibility features described above can be used in conjunction with a keyboard/keyguard to enhance speed, accuracy and the user's overall experience.

The Dynamic Keyboard

Transitioning into the research on text input for people with disabilities, studies have been conducted on users of traditional keyboards who have Parkinson's Disease. Parkinson's is a degenerative neurological disease that causes people to lose some of their motor control and to involuntarily tremor. Because of this, typing on a traditional keyboard can present challenges [Trewin, 2004]. Modern operating systems, such as Windows, Mac OS X, and some flavors of Linux, contain a suite of accessibility features, some of which can be enabled to help people with Parkinson's.

The Dynamic Keyboard is another solution. It consists of software that runs in the background, hidden from the user. It monitors the user's keystrokes, analyzing them to determine which accessibility features would be most appropriate for them [Trewin, 2004]. An example of the algorithm used to dynamically adjust these features follows:

> The key repeat rate is calculated by examining the use of the arrow keys, backspace and delete. Sequences of presses of these keys are observed, including sequences consisting of a single key press that repeats. In each sequence, the user is assumed to be moving towards a target position. The optimum way to achieve this is to hold down the key until the position is reached, then release it. If a user under- or overshoots they may need to make some fine adjustments with extra key presses. If many extra key presses are needed the sequence is considered an over- or undershoot. If few extra key presses are used, the user's positioning was accurate. Over- and undershooting is an indication that the key repeat rate may be too fast. When a user generally over- or undershoots, the key repeat rate is decreased by 0.2 seconds. The idea is that the repeats will gradually become easier to use and not appear to the user to be thrashing between long and short values.
> [Trewin, 2004, 74]

The Dynamic Keyboard automatically adjusts the accessibility features of the operating system to values that it determines are most appropriate but respects feature adjustments that have been made by the user herself. In this way, the Dynamic Keyboard is completely hidden from the user but attempts to adjust features to the user's benefit. A usability study was conducted, and the results provided are very qualitative. Those who typed normally reported that they noticed very little adjustment in settings. This is because the Dynamic Keyboard did not see a need to adjust many settings. Those with Parkinson's reported a mixture of positive and negative responses. Some of the people in this latter group reported that they typed better and more accurately (with fewer errors) when using the Dynamic Keyboard. One user, after typing a nonsense string and then erasing it with the backspace key, reported that her keyboard was not working. This is because the Dynamic Keyboard recommended a high key repeat delay after the nonsense string was deleted and never compensated for it. "Despite the lack of clear benefit from using the program," the author concludes in the results section of the paper, "four [out of 10] participants wanted to continue using it." [Trewin, 2004, 77]

The Canesta Projection Keyboard

We next look at the Canesta Projection Keyboard [Roeber, et. al., 2003] a project targeted at the general population to be used in conjunction with mobile devices. This keyboard consists of a projection of a traditional keyboard on a surface and a sensor which detects keystrokes. This keyboard provides users with a familiar keyboard in a portable and low power manner. The fact that the keyboard is a traditional QWERTY-keyboard means that the user can easily learn this new keyboard. Usability testing was conducted with an adaption of the QWERTY-keyboard which had application keys above the top row and some other modifications. According to the researchers, "the goal of the user studies was to evaluate the Canesta-keyboard relative to other text input devices in terms of error rate, input speed and user satisfaction." [Roeber, et. al., 2003, 713] They comment that, "user satisfaction ratings for the Canesta-keyboard were high for most of the users tested. However, some touch typists encountered text input difficulties and rated satisfaction lower due to the lack of tactile feedback. Additionally, a number of users commented that they would prefer a larger keyboard to the one used in the study." [Roeber, et. al., 2003, 713]

Stylus Input

Another form of text input that is suited for the general population is the use of a stylus. There are many forms of stylus input, from tapping keys on a virtual keyboard, to writing longhand handwriting on a digitized surface, to graffiti and sokgraphs. All of these methods of stylus input assume that the user has a high degree of fine motor control, can hold a stylus, and can accurately tap it and/or draw on the digitized surface.

For someone who can touch type on a standard keyboard, most stylus-based text input methods will not be as fast as touch typing. The memorization of a large number of sokgraphs, however, may lead to typing speeds that rival the best touch typists. For someone who may only have control in one hand or arm but has enough control in that hand to hold a stylus and accurately tap it and/or draw on a digitized surface, stylus input may prove to be a viable input method. In addition, it is not necessary that the user holds the stylus with her hand. If a user does not have enough control in either hand but has sufficient control of another body part to be able to use a stylus, the stylus can be attached to that body part.

Two studies will be discussed that relate to stylus-based input. As is common knowledge, stylus-based input involves tapping or stroking a stylus across a digitized surface. This is most common in the mobile computing context but can be used in desktop applications as well. The first project that will be discussed is Unipad: Single Stroke Text Entry With Language-based Acceleration [MacKenzie, et. al., 2006]. In Unipad, each character consists of a unique stroke or motion of the stylus. This minimizes the movement required. Successive strokes can occur anywhere on the digitized surface, including on top of preceding strokes. Unipad also incorporates word prediction into its interface. The word prediction list is superimposed over the area where letters can be drawn, further reducing stylus movement. The researchers used a normal unigram-based word prediction system with a few modifications. First, they display a list of five words, starting after the first letter is drawn. They also do not sort the list by probability but by size and then lexically with the hope to reduce visual scan time and effort. The system also uses suffix completion. Users add suffixes to a word by making a stroke that enters the suffix mode. They then make a selection from a list of twelve suffixes. This has two benefits. First, it reduces stylus strokes overall. Second, words with multiple suffixes are removed from the word completion list, creating a richer list. They also use a list of the most frequent words at the beginning of a word when no letters are present.

The researchers conducted a user study to analyze keystrokes per character (KSPC). Obviously, a KSPC of 1 means that the user was drawing every character and was not taking advantage of the word prediction or suffix completion features. Analyzing the results of the study, the researchers concluded that, "clearly, learning is important as well. With practice a user's KSPC may fall as features are learned and exploited at the earliest opportunity. And finally, it is not clear that reductions in KSPC yield a corresponding increase in throughput, since the acceleration features also add cognitive demands to the interaction." [MacKenzie, et. al., 2006, 81]

The second stylus-based input method we discuss is the SHARK^2: A Large Vocabulary Shorthand Writing System for Pen-based Computers [Kristensson, et. al., 2004]. The goal of this project is to seamlessly transition users from a stylus keyboard, where each key has its own box that can be tapped with a stylus, to a method of input where users draw a sokgraph - a sequence of vectors between the letters on the stylus keyboard that make up the word. SHARK^2 enables users to draw these sokgraphs anywhere on the digitized surface (they do not necessarily need to coincide with the actual position of the keys on the stylus keyboard). This way, the user can learn the feel of typing a word with a sokgraph and then type it without looking for precise locations of each key. Algorithms were developed to interpret these sokgraphs and to match words with sokgraphs that closely resemble the word but were slightly off.

In this project especially, the learning curve is quite steep in order to gain real benefits, due to the fact that the user must physically memorize many different sokgraphs before she can truly take advantage of the sokgraph typing system. To test what was theoretically possible given a large amount of memorization, the authors of this paper conducted a study on themselves, during which they memorized a severely limited number of sokgraphs and wrote sentences repeatedly. The authors achieved between 69 and 85.6 words per minute, where the "expert performance using an optimized stylus keyboard has been theoretically estimated to about 45 wpm." [Kristensson, et. al., 2004, 51] In this project, as is the case in a number of projects described in this thesis, no usability tests could be found on people with disabilities.

Voice Recognition

Voice recognition is another text input method that is suited for the general population. Growing numbers of people are using commercial voice recognition systems to input text into the computer, as many prefer this input method to typing on a traditional keyboard. Historically, a significant amount of "training," where the computer would listen to the user read a predetermined text and learn the nuances of their speech patterns, was required, however, modern voice recognition systems require less training than older ones do. A condition for being able to use and benefit from voice recognition is the ability to have clear and consistent speech. For people who do not have the fine motor control required to type on a standard keyboard or use a stylus but do have clear and consistent speech patterns, voice recognition may be a viable text input method.

In voice recognition systems, a key element of successful use is the ability to error correct. Error correction can be done from within the voice recognition system or it can be done using another mode of text input. The degree of initial accuracy of the voice recognition system and the ease with which the user can error correct both contribute to the overall usability of this input method for a given user. We will discuss a multimodal system of text input that utilizes voice recognition later in this section.

Eye Gaze Recognition

Eye gaze recognition is a text input method that is specifically targeted for use by people who lack enough control of their hands and arms to be able to type on a standard or adapted keyboard. As stated, eye gaze recognition involves the user looking at a monitor and infrared sensors that detect where the user is looking. In most eye gaze recognition systems, the position of the eyes on the screen corresponds informally to the movement of a mouse, and a user can type by clicking the mouse on an on-screen keyboard. Using infrared technology, researchers can accurately pinpoint where on the screen a user is looking. There are two common ways to activate a mouse click in an eye gaze recognition system. The first is to have one or more switches positioned at areas of the user's body where they have most control. They can hit the switch(es) while they are looking at the desired target (key), and that will produce a mouse click (or type a letter). This has the disadvantage that, since human eyes move relatively quickly, it has been documented that users will inadvertently look away from the target before they actually hit the switch [Zhao, et. al., 2012]. The second dominant way of activating a mouse click in an eye gaze recognition system is that of dwell time. For this, a user stares at a target for a set amount of time to select it. Two disadvantages are present when dwell time is used to activate a mouse click. First, there is an inherent limit to the speed with which a user can type, given that each key press takes as long as the dwell time. Second, it has been documented that error rates are high with dwell time, because users will inadvertently look at an object on the screen for the dwell time period without meaning to select it [Zhao, et. al., 2012].

Traditional eye gaze keyboards use dwell time to have the user activate buttons. In one study [Zhao, et. al., 2012], however, researchers incorporated the use of a tooth clicker - a device that sits on the ear and detects gentle clenching of the jaw. It can differentiate this clenching from normal clenching of eating or speaking. The researchers designed a system where a user could look at keys on an on-screen keyboard and then use the tooth clicker to select keys. They conducted a usability study in which they compared this form of input to traditional eye gaze and dwell time input. The results of the study showed that the shorter dwell time (490 ms) produced input speeds that were higher than the speeds obtained with the tooth clicker but lower than the speeds obtained with the longer dwell time (880 ms). The researchers noticed that the accuracy of the tooth clicker was decreased when a user would look away from the desired key before she would clench her jaw. This was apparently a common phenomenon. The researchers hypothesized, however, that a tooth clicker may be appropriate for applications where the necessary rate of activating objects on the screen was minimal, such as surfing the web, rather than typing a document.

Sign Language Recognition

For users who know sign language and use it on a regular basis, automatic sign language recognition may be a viable form of text input. Sign language recognition consists of one or more cameras that track the user's hands and/or sensitized gloves that track the position and motion of the hands. In either case, the computer translates signs into text, using hidden Markov models to predict what the user is saying. A problem when recognizing American Sign Language (ASL) is that the syntax and phrasing of ASL does not directly match that of spoken English. Furthermore, as Starner, Weaver and Pentland note, "conversants in ASL may describe a person, place, or thing and then point to a place in space to store that object temporarily for later reference." [Starner, 1998, 1371] This, along with facial expressions, which are used heavily in sign language, are not able to be parsed in modern sign language recognition systems.

Sign language recognition may be a viable form of text entry for those who are familiar with the gesture based language. As mentioned, sign language recognition converts a user's signs into text by tracking the motions of their hands, either with cameras or sensitized gloves. In one study, researchers investigated different accuracy rates based on where the camera was placed. In the first experiment of the study, the camera was placed on a desk pointed at the user, in a position of someone to whom the user would be communicating. In the second experiment, the camera was embedded in the visor of a cap worn by the user herself. The camera pointed down from the visor towards the user's hands to record the perspective of the user herself.

In this particular study, a limited 40-word vocabulary was used. Word accuracy of about 90% was reported for the desk based system after extensive training, and word accuracy of about 97% was reported for the cap based system. These results, of course, are not indicative of how the system in this study would respond to a larger vocabulary. Furthermore, the camera image was highly controlled to maintain continuity between images. Changes in image background can offset the accuracy of hand placement recognition, and occlusion of one or both hands was a major problem discussed.

Nonetheless, the authors believe that sign language recognition systems can be a viable text input method. They also propose that the visor based system, together with a wearable computer, could become a robust system for ASL to synthesized speech conversion [Starner, 1998].

Switch Input

Switch activation is another input method that is traditionally targeted for use by people with physical disabilities, but elements of it are used in mobile applications as well. There are two distinct forms of switch input: single switch and multi switch input. For single switch input, a user may only have enough control to press one pushbutton switch. In this situation, software displays a list or a two-dimensional matrix of items from which the user can select. In the case of a list, the software repeatedly scans through the list, pausing at each item for a set amount of time. The user selects the desired item by hitting the switch when the software pauses on that item. In the case of a two-dimensional matrix, the software cycles through the rows (or columns) of the matrix until receiving a switch-hit from the user. Then, the software scans through the items in that row (or column) until receiving a second switch-hit from the user. In either case, even though the user only needs to be able to press a single switch, she needs enough control to be able to press it while the software is pausing on the desired item. Increasing the amount of time the software pauses on each item will increase accuracy but decrease overall speed. As is obvious, the letters, numbers, punctuation, and other keys on a standard keyboard can comprise the list or matrix from which a user can select. The single switch input method is generally very slow, as the user must wait for the software to scan through each item each time she wants to select an item.

The speed of the switch input method can be increased if the user is able to accurately hit more than one switch. For the one-dimensional list and two-dimensional matrix, if the user is able to hit two switches, one switch can control the scanning, and the other can be a selector. A user can quickly cycle through the list or rows (or columns) by holding down the first switch and releasing it at or before the desired item (or row). Repeatedly hitting the first switch will cause the software to cycle through the items. In this way, the user is not waiting for the software to scan, pausing at each item for a set period of time. As stated, the second switch can serve as the selector.

If the user can accurately hit three switches, one of them can be a selector, one of them can move the focus down, and the other can move the focus up. This, again, will increase overall speed, as the user has more control over where the focus is. Having four switches does not increase input speed when selecting from a one-dimensional list or two-dimensional matrix, but having five switches dramatically increases speed when using a two-dimensional matrix. Four of the switches can be used to control the focus, each controlling a different cardinal direction, and the fifth switch can be the selector.

There is another paradigm for switch input, other than using switches to select from a one-dimensional list or two-dimensional matrix. This paradigm requires at least two switches (or one switch where the user can differentiate between long and short switch presses). In this paradigm, sequences of switch presses correspond to letters, numbers, punctuation, modifier keys, etc. The user presses sequences of switches to type. In this paradigm, as was the case with the first paradigm, increasing the number of switches will increase overall speed. The user must be able to memorize the sequences, or have a visual display that can guide her in her choosing.

Dit4Dah and BinScroll

Two projects are worth mentioning as they each take a separate approach to switch input. The first is Dit4dah [Tanaka-Ishii, et. al., 2004]. This project incorporates Morse Code based text entry with a word prediction system that takes into account the difficulty of entering a given word. A user can input text using long and short switch presses, modelled after Morse Code. When a word is being typed, a list of potential completed words is presented to the user. The user can either continue entering letters using Morse Code or depress the switch for an extended amount of time to scroll through the list presented. The authors claim to be the first ones to have incorporated entry difficulty in the selection of the word prediction list.

The second project that relies exclusively on switch input is BinScroll [Lehikoinen, et. al., 2002]. While not directly related to text entry, BinScroll is included here as the project contains aspects that will be useful in our analysis of text input. The objective of BinScroll is, using only four buttons, to quickly select a desired item from an ordered list. This is accomplished through a process that is based on the binary search algorithm, with a few additions. The project was designed mainly for mobile applications, but it also has implications for users with disabilities. Also, the ordered list could be replaced by an alphabet and other keyboard keys, enabling a user to input text.

Virtual Logo Keyboard

There are many other research projects and commercial products that make use of the scanning method and switch input. One such project that is of particular interest to computer science educators is the Virtual Logo Keyboard [Norte, et. al., 2007]. This was designed to aid people with disabilities in the act of writing computer programs in the Logo programming language. The virtual keyboard consists of the traditional keys of a standard keyboard, but it also has keys for common Logo commands. The Virtual Logo Keyboard can either interact directly with a Logo interpreter, issuing commands directly to it, or send its output to a text editor.

A usability study of the Virtual Logo Keyboard was conducted on three high school students who were in the process of learning the Logo programming language. Two of the students had physical disabilities while one did not. The study concluded that the virtual keyboard was not beneficial for the student who did not have a disability, although they did benefit from use of the integrated help system. For both of the students with disabilities, the Virtual Logo Keyboard was helpful. For the student who was able to use a mouse to interact with the keyboard, the input speed greatly increased. It was observed that the input speed did not increase as much for the student who needed to use the scanning method, but the fact that the Virtual Logo Keyboard can issue entire commands meant that the overall input time for this student was decreased as compared to input without use of the Logo keyboard [Norte, 2007].

Wii Multimodal System

As is the case with the Virtual Logo Keyboard, many projects designed to help people with disabilities interact with computers incorporate more than one technology. This is the case with the Wii Multimodal System (WiiMS) [Honye, et. al., 2012]. WiiMS consists of a head tracking system to manipulate the position of the cursor on the screen and voice recognition to simulate mouse and keyboard buttons. The project was designed primarily for use by people with spinal cord injuries who are not able to control their body from the neck or shoulders down. An interesting aspect of the voice recognition is that the keyboard and mouse commands are divided into sections, and the user must orally switch sections before she can say a button within the section. This is to attempt to reduce error and ambiguity. Usability tests for this project were conducted on able bodied persons with a wide range of native languages, with the hope of getting an indication of how the system would work with people with disabilities and varying speech patterns. The tests were conducted on these subjects due to the unavailability of people with disabilities. Researchers used the study to analyze the learnability, intuitiveness, user satisfaction, and effectiveness of WiiMS.

Foot Controls

Going back in time, prior to the popularity of graphical user interfaces, studies were conducted on how to decrease the time required to position a cursor in a text editor. Today, we do this with incremental positioning systems such as arrow keys, relative positioning systems such as mice, and absolute positioning systems such as styluses. All of these systems require time to use. One study [Pearson, et. al., 1986] looked at reducing this time for people using workstations. The researchers argue that input time is wasted while the user switches between a keyboard and a mouse or other cursor positioning system. If this wasted time could be reduced, input speeds could increase. They define homing time as the time required to move the hand from the mouse back to the keyboard. As the background to their research, they discuss various other ways to reduce homing time. One way is to embed the cursor positioning system inside the keyboard. Another is to make a one handed keyboard, which frees the other hand for cursor positioning, but the problem with one handed keyboards (as well as sequence based switch input methods) is that seldom used characters are hard to recall.

In their paper, the researchers present four different methods for foot-based cursor positioning. The first approach involves no moving parts. It simply has the foot resting on a flat or otherwise shaped sensitized surface which can detect the foot's movement. The cursor would move in the direction that the foot moved. Similar to picking a mouse up off the desk to continue moving in the same direction, a user could pick her foot up off the surface and place it back down at a different location. The second approach was deemed "the swing approach," where each foot was suspended in the air, each able to swing forward and backward independently of each other. One leg would control the vertical motion of the cursor and the other would control the horizontal motion. Swinging the foot forward would move the cursor one way and backward would move it in the opposite direction. The third approach was similar to the second but each foot would be able to swing as if it was a pendulum. This would create more options. The fourth approach involved placing the toe of each shoe in a rectangular object that had switches on each of the four walls. By moving the foot in specific ways, a user could engage between 0 and 4 of these switches at once, thus, providing many combinations. No usability statistics were provided.

IV. Method Comparison to a Standard Keyboard

In this section, we will first analyze the standard keyboard and then compare the standard keyboard to a select number of the alternative input methods previously outlined. To begin, we note that a standard keyboard is the dominant means of inputting text into a computer. Standard keyboards vary in size, number of keys, thickness of keys, etc., but they all take the same basic shape.

Most people who do not have physical disabilities find the standard keyboard to be easy to learn, easy to use, efficient, and accurate. The time required to learn how to type on a standard keyboard will depend on whether the user has previous experience typing on a typewriter, since the location of the keys on a standard keyboard is based on that of typewriters. This, perhaps, is the biggest reason why standard keyboards have become so popular. They rely on the same basic key layout as typewriters, so historically, people with experience using typewriters have found keyboards to be familiar. This learning time will also depend on the user's ability to understand the basic functions of text input into a computer. Once learned, users generally do not have to think about typing on the standard keyboard, especially if they touch type, and can focus on the text being inputted.

Alternative keyboard layouts have been developed to try and speed up the text input process. The QWERTY key layout was originally developed for typewriters, with the goal of actually slowing down the typing process, since typists were getting too fast for first and second generation mechanical typewriters and keys were sticking. Today, the QWERTY key layout still ships with the vast majority of laptop and desktop computers, although alternative key layouts that enable users to type faster exist. One such layout, which attempts to reduce finger motion by having the most commonly used keys on the home row of the keyboard is the Dvorak Keyboard. [Joyce, et. al., 1990]

That people can easily learn and use standard QWERTY keyboards, given that they are part of the culture that emerged from typewriters, is not always the case for people with disabilities. Depending on the type and magnitude of the disability, the standard keyboard may be hard, or even impossible or impractical, to use. Several reasons account for this. First, the user may lack control of her hands or arms, something that is required to type on a standard keyboard. Even if she has gross motor control of her hands and arms, however, she may lack the fine motor skills necessary to hit isolated keys on the standard keyboard. Her lack of fine motor skills may cause her to press numerous unwanted keys when she is trying to press a certain key.

The Dynamic Keyboard is a piece of assistive technology that was developed to mainly assist people with Parkinson's Disease who progressively lose fine motor abilities in their hands. As stated above, the Dynamic Keyboard adjusts accessibility features on the fly based on an analysis of the user's typing. It assumes that the user is typing on a standard keyboard. Through a suite of algorithms that monitor and analyze the user's typing style, the Dynamic Keyboard determines which accessibility features of the Windows operating system would be most beneficial to them and then adjusts those features automatically, without the user's knowledge. Accessibility features manually set by the user are respected and left unchanged.

As stated, the Dynamic Keyboard was developed for people with Parkinson's Disease, who can have tremors in their hands as well as loss of fine motor control. Many of these users may not have prior experience with the accessibility features of common operating systems, may not know that they exist, and/or may not be interested in adjusting them over a long period of time to achieve the ideal set of settings. The Dynamic Keyboard is a solution to this problem, as it adjusts settings for the user.

Additionally, it has been noted in the research that the symptoms of Parkinson's Disease vary based on the length of time that has elapsed from when the person took Parkinson's medication [Trewin, 2004]. Symptoms intensify as the medication wears off. Because of this, someone with Parkinson's may use a keyboard one way when her medication is strong and another way when it is wearing off. Instead of having to manually adjust accessibility settings throughout this cycle, the Dynamic Keyboard detects changes in keyboard usage and adjusts settings automatically.

Usability tests for the Dynamic Keyboard have only been conducted on people with Parkinson's Disease and people who did not have any disabilities, although people with other disabilities that affect their fine motor control may benefit from the use of the Dynamic Keyboard. Those without disabilities reported seeing little or no change in how the keyboard responded, and this was verified by the fact that the Dynamic Keyboard made very few accessibility settings adjustments [Trewin, 2004]. For these users, the Dynamic Keyboard detected standard typing with relatively few errors, so it did not result in a need to alter settings much. This suggests that while people without disabilities would not find much benefit in the use of the Dynamic Keyboard, it does not interfere with their typing as it acts much like a standard keyboard.

As mentioned above, the Canesta Projection Keyboard was developed to be used in conjunction with mobile devices. It projects an image of a standard keyboard on a surface and uses sensors to detect finger motions. For able-bodied users who are somewhat familiar with a standard keyboard, the Canesta Projection Keyboard is easy to learn and easy to use, as documented in the user study. For touch typists, this keyboard presents an obstacle, since touch typists are used to resting their fingers on the home keys, something that interferes with the Canesta Keyboard's sensors. For people who already type on traditional keyboards, the Canesta Keyboard would be easy to learn, since it is based on the well-known QWERTY layout.

The Canesta Projection Keyboard does not possess any significant hardware or software features that would be beneficial to people with physical disabilities, particularly those who cannot use a traditional keyboard. Given that the keys on the Canesta Projection Keyboard are merely projected on a surface, there is no tactile feedback of where the keys are or the barriers between one key and another. For people who do not have a significant amount of fine motor control, this could create a barrier. Furthermore, since the intended use of the Canesta Keyboard is in mobile applications, depending on the disability, the user herself may not be able to physically set the keyboard up in mobile circumstances.

Our analysis of the Unipad system of stylus input begins with noting the fact that each character consists of a single stylus stroke. The shapes of these strokes were designed to mirror traditional upper and lower case letters, with a few modifications that make each character distinct and easy to draw [MacKenzie, et. al., 2006]. A user will have to learn each of these shapes before she can use Unipad efficiently. Once learned, however, drawing shapes for each letter will come naturally to the user and she will not have to think about each character. Reductions in KSPC can be achieved by using the word prediction element of Unipad, but as the researchers noted, this does not necessarily increase overall throughput, since scanning the word prediction list takes time and requires more cognition.

The main two challenges faced by people with disabilities who are trying to use Unipad is whether they will be able to use the stylus and whether the system can recognize their strokes to a sufficient degree of accuracy. If these two challenges are overcome, Unipad could be a beneficial input method for people with disabilities, and especially for people who may not be able to use a traditional keyboard. If the user can accurately use a stylus but uses it very slowly, the word and suffix completion features of Unipad will be of huge benefit, as they will increase speed.

Our analysis of SHARK^2 is similar to that of Unipad, except we note that for SHARK^2, the learning curve is quite high, since a user must learn a large number of sokgraphs before she can truly take advantage of the technology. As noted above, however, once a sufficient number of sokgraphs are learned, a user's typing speed can increase dramatically, perhaps even rivaling speeds of touch typists on traditional keyboards.

People with disabilities would face similar challenges in using SHARK^2 as they would face when using Unipad. In addition, a user must be accurate enough in her use of a stylus to draw the sokgraph in such a way that the system will recognize them as words. If a user with a disability can draw sokgraphs with sufficient accuracy, this system may be more beneficial than a traditional stylus-based key tapping system, as each sokgraph can be drawn in one long stroke, without having to lift the stylus off the digitized surface. This may speed up input for people for whom lifting the stylus takes time.

The foot based input methods described above create opportunity for interesting analysis. Though the study was targeted at cursor positioning, the concepts can also be applied to mouse positioning and to typing itself. It is clear that the first approach described in the study (that of placing the foot on a flat or otherwise shaped sensitized surface) is ideal for cursor and mouse positioning and may not be suitable for text entry, as it would be difficult to divide up the surface into different sections in order to distinguish between keys or switches. That being said, the other ideas discussed in the paper can be applied to text input. All of these ideas have discrete switches that are engaged when the user moves her foot or leg in certain ways. Combinations or sequences of these switch presses could represent keys on a traditional keyboard, characters in the ASCII table, or sequences of characters and/or entire words. For people who cannot use their hands but who have sufficient foot control, this may be a great option. The final idea presented in the paper (*i.e.,* having four switches controlled by each foot) could yield a powerful system of input for people with disabilities, based on the switch input methods discussed above.

The following input methods were all developed for people with some type of physical disability. We will analyze them using the same criteria we have been using but we will focus on how each method benefits people with disabilities. When appropriate, we will also mention how the method could be used by people without disabilities and how it compares to a standard keyboard.

The first such method we will analyze is Dit4dah. As mentioned above, Dit4dah uses a Morse code based system of switch input text entry. Users hit a single switch for a long or short press to indicate bits of Morse code. A word prediction list is also displayed to the user, and she can select words from it with an extended press of the switch. Obviously, entering text with a single switch through Morse code will be slow, even with word prediction, since a user must scan through the lists of words in order to select one. Input speeds will be significantly slower than a standard user of a traditional keyboard, but for users who can only operate a single switch, Dit4dah possesses some advantages over traditional scanning input methods. Learning the Morse codes for each character will demand a significant amount of time, but once learned, the user may be able to type at a faster rate than if she was using the scanning method.

The other method that uses switch input which we analyze is BinScroll. Through BinScroll, a user can select an item from an ordered list with only four buttons, using a modified version of the binary search algorithm. BinScroll was originally developed and tested with a large list of movie titles, where the user was tasked to select a specific title from the list [Lehikoinen, et. al., 2002]. However, as noted above, this list could easily be adapted to include letters, other characters, and possibly common words and phrases. The binary search algorithm provides a method to select the desired result in log n time, where n is the length of the list. If a user has the ability to control four switches, BinScroll, adapted to contain letters, symbols, and perhaps common words, could be beneficial, as it could create a quick input method, based on binary search.

The results of the usability study of the Wii Multimodal System (in which a user controls the mouse by head movements and types on a keyboard through voice commands) reveal that users found the system easy to learn, easy to use, and intuitive [Honye, et. al., 2012]. Study participants were more comfortable with the cursor positioning system than they were with the text input method, since this method only recognized letters and other keyboard keys instead of entire words. They found this to be too slow for efficient text input. They also reported that the system would misrecognize letters that sound the same, such as “m” and “n”. Though the research was targeted for people with spinal cord injuries, study participants (all able-bodied people) reported that people without disabilities may want to use the cursor positioning system, as they found it to be easy to use and efficient.

V. The Framework

We will now propose a framework for the development of a recommender system that will be able to suggest a ranked order of input devices based on a user’s specific abilities. Our framework aims to be inclusive in two ways. First, we aim that input methods that span a wide spectrum will all be able to fit in our framework. We acknowledge that it would be impossible and impractical for us to include all types of input methods here, but we hope that, based on the proposals and analysis here, this framework will allow others to analyze other forms of input that either exist now or are yet to be invented. Second, it is our intent that our framework thoughtfully and respectfully encompasses persons with a wide array of physical abilities. We create this framework with the hope that it can be used in future research as a basis for a recommender system to enable a person with a specific set of physical abilities to determine what type/form/method of text input would be most appropriate for them to use. Developing a recommender system is beyond the scope of this thesis, primarily because the amount of data required is too large and much of it is not available, but we hope that this may serve as a foundation. Limitations other than physical can be included or added to the framework, but for now we will concentrate on physical issues.

In designing a framework to assist a user in determining the input method/device that would be most suited to her specific set of abilities and challenges, it might be natural to ask a series of questions that the user could answer, after which a ranked list of recommended methods/devices would be produced. Examples of such questions follow in no specific order.

- Any fine motor control in either hand?
- Differentiate finger presses?
- Hit more than one keyboard key at a time?
- When trying to use standard keyboard, constantly hit undesired keys or unable to hit desired keys?

- When trying to use standard keyboard, observe undesired multiple transmissions of same character?
- When trying to use standard keyboard, requires much time and adjustment to accurately position cursor using the arrow keys?
- Hold a stylus?
- Accurately draw and tap on a digitized surface?
- Memorize stylus handwriting?
- Memorize sokgraphs?
- Hit four or more switches positioned at any location of the user's body?
- Hit one - three switches positioned at any location of the user's body?
- Control timing and duration of switch presses?
- Rate of character input slow enough to warrant word prediction?
- Clear and consistent speech pattern?
- Control of eye movement?
- Memorize switch encoding scheme?

Though these questions seem like the natural questions to ask when making a determination of the most appropriate input device, we found that these questions are not, in fact, helpful when developing a recommender system. To ask these questions implies that we construct our framework of questions in the form of a tree or directed acyclic graph (DAG). If we were going to construct our own framework in the form of a tree or a DAG, we would put a question about a user's abilities at each node. These questions could be as broad as whether they have any control of their extremities or as specific as whether they have enough fine motor control in all ten fingers to accurately touch type on a standard keyboard. The questions (nodes) would be placed in the tree or DAG in such a way that one question would lead to the next and eventually the path would lead to a leaf node, which would be a recommended input method or set of methods.

This approach, however, is extremely rigid, limiting, and prescriptive. In a tree, by definition, there is exactly one path from the root to any one leaf. The manner in which a user would answer the series of questions in a tree (or in a DAG in this case) would automatically exclude certain types of input devices that might, in fact, benefit the user. We, therefore, do not believe that a framework of questions structured as a tree or a DAG is an appropriate method for this application.

Benefits of a Recommender System

We believe producing a ranked list of recommended input methods/devices is the most appropriate way to construct this framework, given that we are dealing with human beings who are somewhat unpredictable. No matter how detailed the questions are and no matter how much thought and effort goes into designing appropriate and targeted questions that are hoped to lead a user to the ideal method/device, no framework will fully encapsulate the nuances that emerge when dealing with human beings. These nuances are perhaps more extreme and less well documented for people with disabilities than they are for the general population, so this adds yet another layer of complexity. Therefore, we believe that a framework which leads a user to a single answer (one specific input method, for example) is inappropriate because it may very well be the case that that input method may not work or may not be ideal for that specific user. Hence, a recommender system, producing a ranked list of suggested input methods/devices, is much more appropriate to this application. It gives the user the ability to research and then perhaps try the top ranked suggestion, and then if that does not work, continue down the list to the next suggestion.

A Recommender System for Suggesting Input Devices

We will now propose the framework for the development of a recommender system for suggesting a ranked order of input devices based on a user's disabilities. We construct this recommender system in the form of a matrix. Along one axis (the x axis in our model), we have three categories of columns. In the first category, we have four columns which represent a person with the standard set of abilities. These columns are Standard Speech, Standard Gross Motor, Standard Fine Motor in Hands, Standard Sight. Our reasoning for dividing a person with the standard set of abilities into multiple columns (we chose four) will be evident shortly. Our second category includes columns for No Speech, No Gross Motor, No Fine Motor in Hands, No Sight. We will explain these columns shortly, as well. Our third category includes only one column, Increased Fine Motor in Feet, but it can be expanded to include other columns. This third category requires special attention and will be explained at the end of this matrix description. Along the other axis of our matrix (the y axis in our model), we place input devices.

	Standard Speech	Standard Gross Motor	Standard Fine Motor in Hands	Standard Sight	No Speech	No Gross Motor	No Fine Motor in Hands	No Sight	Increased Fine Motor in Feet (enter 1 for very good)
Standard Keyboard			0.8				-0.8		0.5
Software Adaptations			0.4				-0.3		0.6
Keyguard			0.02				0.4		0.7
Stylus Recognition			0.5					-0.8	0.9
Voice Recognition	0.7				-1	0.9		0.7	
Eye Gaze Recognition				0.03		0.1		-1	
Sign Language Recognition			0.01						
Switch Input		0.1	0.1			0.01	0.7		0.8

Table 1

We produce the recommended ordering of input devices by multiplying this matrix by a column vector which represents a person. This column vector has cells that represent each column in our matrix, and in each cell, we place a value between 0 and 1. These values represent the degree of applicability of that column in the person we are evaluating. For example, a person with the standard set of abilities would have 1's in the first four cells of the column and 0's in the second four cells. A person with no fine motor control in their hands would receive a 0 for the Standard Fine Motor in Hands column but a 1 in the No Fine Motor in Hands column. Gradient values in the person vector are permitted, so long as the values of pairs of cells which correspond to each other (i.e., Standard Speech and No Speech) sum to 1. We will leave the third category of columns in our matrix for the end of this description.

Determining the appropriate values for the matrix itself is beyond the scope of this thesis. We acknowledge that the data we use to demonstrate the effectiveness of our framework are largely speculative, but this is unavoidable. The data, however, are based on an analysis of the research presented in earlier sections of this thesis, as well as on the prior experiences of the author, who has a physical disability. As stated, though we lack accurate and sufficient data to implement such a recommender system, we propose the framework with the hope that it can be a catalyst for future research and implementation.

We will describe the data we use category by category, and by doing so, we hope to make our theory clear. We will begin with the first category of columns - the category which represents a person with the standard set of abilities. As stated, the values in the person vector that correspond to this first category would be all 1's for a person with the standard set of abilities, and all values in the person vector which represent the second category would be 0. Thus, for a person with the standard set of abilities, only the values in the first category of columns in our matrix influence the result of the dot product multiplication. This is a key insight on which we base much theory.

For a person with the standard set of abilities, we encode the desired ranked ordering of input devices by setting values in our matrix in the first category of columns in the following manner. All values in these first four columns are between 0 and 1 in our implementation, with higher numbers representing higher recommendation scores. For example, we gave the Standard Keyboard a score of 0.8 and Voice Recognition a score of 0.7, but we gave Sign Language Recognition a score of 0.01, since we assume that a person with the standard set of abilities would have extremely little benefit from the use of sign language recognition. It is important to note in which column we place these values. We place values for Standard Keyboard, Keyguard, and Stylus Recognition in the Standard Fine Motor in Hands column, since the usefulness of these input methods only depends on the degree of fine motor ability that the person has in her hands. Similarly, we place the value for Voice Recognition in the Standard Speech column, since the usefulness of voice recognition only depends on the degree that the person can speak normally. For some of our other input methods (Switch Input, for example), we distribute the desired value across multiple columns. A person can have fine or gross motor ability (or both) to have switch input be useful, so we distribute the desired value for Switch Input across the two appropriate columns. We are cognizant, however, that we want to maintain the desired ranked output for a person with the standard set of abilities by having the sums of each row correspond to the desired scores for each input method.

Next, we will explain the values in the second category of columns. Each column in this category corresponds to a negation of a column in the first category, although the values in the columns in the second category are somewhat different from those in the first category. All values in this second category range from -1 to 1, with -1 meaning that the input device would be extremely unuseful for someone with the given challenge (No Fine Motor in Hands for the Standard Keyboard, for example), and 1 meaning that the input method would be very helpful. As was the case with the first category of columns, we can either place values in only one column for a given row or in multiple columns. An example of an input device that has values in multiple columns is Eye Gaze Recognition, where we give it 0.1 in the No Gross Motor column and -1 in the No Sight column. It is important to note that as opposed to the first category where we only placed values in columns where the ability directly impacted the usefulness of a given input device, here in the second category, we have a bit more leeway. This, in fact, is the power of the recommender system. We are able to suggest that someone who has no gross motor ability may want to try eye gaze recognition by giving a positive value in that cell.

Now we shall explain the third category of columns. As was stated, there is only one column in this category in our implementation, but this category, as well as the other two, can be expanded. The column we have in this third category is Increased Fine Motor in Feet. We put this column for someone who has developed a degree of fine motor ability in their feet that is above what the average person would have. Most likely, this compensation would be a result of a lack of fine motor ability in the hands, and this fact must be accounted for in the dot product multiplication, because otherwise the math does not work. A 0 in the person vector for this column means that the person has average fine motor ability in their feet, and increasing scores mean increased ability. However, since the person is presumably compensating for a lack of fine motor ability in their hands, the value in the person vector that corresponds to No Fine Motor Ability in Hands must be reduced proportionately before the multiplication is executed in order for the math to work.

Implementing the Recommender System on Sample Users

We will now present a few examples to illustrate how this recommender system could function. It is important to bear in mind that the values in our matrix are largely speculative and that better results would be yielded from a larger table with researched data. We present these examples, however, to illustrate the method, rather than specific rankings.

We will consider four hypothetical individuals. We will say that Person A does not have any noticeable disabilities. We will say that Person B has athetoid cerebral palsy, Person C does not have any control in their legs, has tremors in their hands, and has a very mild speech impediment, and Person D is blind and has amputated arms but has compensated for lack of fine motor control in hands by developing very high fine motor control in the feet. Here are the person vectors for these four individuals:

	Person A	Person B	Person C	Person D
Standard Speech	1	0.7	0.9	1
Standard Gross Motor	1	1	0.5	0.5
Strd F. Motor Hands	1	0.4	0.7	0
Standard Sight	1	1	1	0
No Speech	0	0.3	0.1	0
No Gross Motor	0	0	0.5	0.5
No Fine Motor Hands	0	0.6	0.3	1*
No Sight	0	0	0	1
↑ Fine Motor Feet	0	0	0	1

*This becomes 0 before we multiply.

We will now present the results of the evaluations of these four hypothetical individuals. We will include the ranked ordering for each person as well as the score that each input method received as a result of the dot product multiplication.

Person A:

Standard Keyboard	0.8
Voice Recognition	0.7
Stylus Recognition	0.5
Software Adaptations	0.4
Switch Input	0.2
Eye Gaze Recognition	0.03
Keyguard	0.02
Sign Language Recognition	0.01

Person B:

Switch Input	0.62
Keyguard	0.248
Stylus Recognition	0.2
Voice Recognition	0.19
Eye Gaze Recognition	0.03
Sign Language Recognition	0.004
Software Adaptations	-0.02
Standard Keyboard	-0.16

Person C:

Voice Recognition	0.98
Stylus Recognition	0.35
Standard Keyboard	0.32
Switch Input	0.315
Software Adaptations	0.19
Keyguard	0.134
Eye Gaze Recognition 0.08	
Sign Language Recognition	0.007

Person D:

Voice Recognition	1.85
Switch Input	0.905
Keyguard	0.7
Software Adaptations	0.6
Standard Keyboard	0.5
Stylus Recognition	0.1
Sign Language Recognition	0
Eye Gaze Recognition	-0.95

VI. Conclusion:

As is evident, the results of the recommender system for the four hypothetical individuals presented are not ideal. As stated, this is due to the fact that the data in our matrix are largely speculative. We are confident, however, that our method for determining the rankings is sound. Much of the data required for an accurate matrix does not exist, so it is our hope that this thesis may serve as a catalyst for further research in this area. Once accurate data is available, we believe that the framework we have presented for the recommender system can be implemented and used widely.

We have also presented a survey of text input devices and have formulated hypotheses as to which input method would be most suited to which population. We have explored input methods that span a wide spectrum, from the standard unmodified keyboard to voice, eye gaze, and sign language recognition, and from keyguards and switch input to stylus and handwriting detection. We have documented the advantages and drawbacks of each method investigated and have tried to draw connections between methods where appropriate. Knowledge is power, and we hope that this thesis will serve as a foundation of knowledge, comparing text input methods to each other and proposing a robust system to recommend devices based on ability.

VII. References

Americans With Disabilities Act of 1990, US Pub. L. No. 101-336, 104 Stat. 328 (1990).

Honye, S., Thinyane, H. WiiMS: Simulating mouse and keyboard for motor-impaired users. In *Proceedings of the South African Institute for Computer Scientists and Information Technologists Conference.* October 2012, 188-195.

Ivory, M., Hearst, M. The State of the Art in Automating Usability Evaluation of User Interfaces. *ACM Computing Surveys (CSUR),* 33, 4 (December 2001), 470-516.

Joyce, B., Moxley, R. Comparing the use of the Dvorak and QWERTY. *Journal of Computing in Childhood Education*, 1, 4 (August 1990), 35-45.

Kristensson, P., Zhai, S. SHARK^2: A Large Vocabulary Shorthand Writing System for Pen-based Computers. In *Proceedings of the 17th annual ACM symposium on User interface software and technology.* October 2004, 43-52.

Lehikoinen, J., Salminen, I. An Empirical and Theoretical Evaluation of BinScroll: A Rapid Selection Technique for Alphanumeric Lists. *Personal and Ubiquitous Computing,* 6, 2 (April 2002), 141-150.

MacKenzie, I., Chen, J., Oniszczak, A. Unipad: Single Stroke Text Entry With Language-based Acceleration. In *Proceedings of the 4th Nordic conference on Human-computer interaction: changing roles.* October 2006, 78-85.

Norte, S., Lobo, F. A Virtual Logo Keyboard for People with Motor Disabilities. In *Proceedings of the 12th annual SIGCSE conference on Innovation and Technology in computer science education.* 39, 3 (September 2007), 111-115.

Pearson, G., Weiser, M. Of Moles and Men: The Design of Foot Controls for Workstations. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. New York, NY, April 1986, 333-339.

Roeber, H., Bacus, J., Tomasi, C. Typing in Thin Air: The Canesta Projection Keyboard - A New Method of Interaction with Electronic Devices. In *Extended Abstracts on Human Factors in Computing Systems.* April 2003, 712-713.

Starner, T., Weaver, J., Pentland, A. Real-Time American Sign Language Recognition Using Desk and Wearable Computer Based Video. *IEEE Transactions on Pattern Analysis and Machine Intelligence*. 20, 12 (December 1998), 1371-1375.

Tanaka-Ishii, K., Frank, I. Dit4dah: Predictive Pruning for Morse Code Text Entry. In *Natural Language Processing - IJCNLP 2004*. 3248 (2005) 765-775.

Trewin, S. Automating Accessibility: The Dynamic Keyboard. *ASSETS'04*, 2004, 71-78.

Zhao, X., et. al. Typing with Eye-Gaze and Tooth-Clicks. In *Proceedings of the Symposium on Eye Tracking Research and Applications.* March 2012, 341-344.