**Assessing Kinesthetic Finger Pinching Performance in a Text Entry Setting**

M. Sc. Thesis Proposal  
July 09th, 2012  
Bartosz Bajer

### 1 Introduction

The human body has the natural ability of knowing the physical location and orientation of itself and its parts. This ability, known as kinesthesis, is advanced enough that it allows us to learn and perform actions that require little to no visual and mental attention. We utilize kinesthesis when performing tasks in environments that are static and this helps us develop kinesthetic (muscle) memory. For instance a person who has learned to drive a manual car does not need to look or think about shifting gear nor accelerating and breaking. The body simply reacts with the required action as soon as it is necessary because through practice it knows where the objects are and the forces required to actuate them. The highest visual and mental attention required when learning such a task is during training. After training, as long as the environment doesn’t change, the body can continue to perform tasks using kinesthetic memory with little or no visual or mental attention. There are instances where visual attention and training are not required. Such is the cases when the body uses the kinesthetic sense on itself. With eyes closed one can easily touch the tip of the index finger on the left hand to the tip of the ring finger on the right hand. Similar tests are performed by law enforcement to assess an individual’s sobriety, such as closing your eyes and touching your nose with the tip of your index finger. In these cases, performance of kinesthetic actions also relies highly on haptic feedback. The driver receives queues from the forces that shifting gears exert through the medium of the stick. Similarly we know we have completed the finger touching task without seeing it because of the forces the fingers produce against each other. More complex actions can be learned as well. Take for example a guitarist; after sufficient practice such a person would appear to perform any musical note on their guitar effortlessly due to honed muscle memory. It is not hard to imagine cases where we take advantage of the natural human talent to learn and perform highly precise body movements in computer interactions.

Similar to the example of driving a manual car, static environmental kinesthetic interaction is seen when we use a computer. In order to interact with a desktop computer, a person often sits at a desk on which there is a keyboard, a mouse, and a monitor. The environment is static in that the keyboard, mouse and monitor are always in the same place. In this way the body is able to adapt to the arrangement such that after sufficient training the person can interact with the environment using kinesthetic memory. This allows typing performance to develop such that anexperienced typist simply thinks of a word and their fingers immediately react by pressing the keys on a keyboard.

We have become highly accustomed to static environment kinesthetic interaction with computers. These same static environment kinesthetic interaction principals are used when designing mobile hardware and software environments in the hope that our kinesthetic abilities will naturally translate. This is not ideal because often time’s people try to perform other tasks when interacting with their mobile device, but the device itself requires two hands and visual attention to operate. For instance one can drive and talk on their phone at the same time because talking does not require visual or physical attention. But texting is not ideal because texting interfaces have been design to require visual attention and our kinesthetic sense requires users to hold the device in a specific orientation when typing. The interface of a mobile phone requires the users attention at all times for tasks like application selection, interface manipulation, and text entry. For this reason many different governments have banned the use of phones while driving, as they are inclined to distract motorists. The underlying problem is that mobile devices are designed to be held and oriented in a way that users have attuned their kinesthetic sense too it. This problem may be mitigated if interfaces were designed to utilize the kinesthetic interactions against one’s own body. The human body as an interface has potential since its local configuration and physical orientation is always relative to itself and doesn’t change. Its flexibility allows for the performance of numerous discrete actions.

In order to be able to design a proper body interface one will have to measure bodily performance in completing interactive tasks. Recently we have seen kinesthetic interaction with devices in gaming consoles such as the aptly named Microsoft *Kinect*. This technology was designed to read head, arm, and leg movement through inferred cameras with the intention of using the body as a video game controller. This technology has worked well for games featuring physical activity such as dance games, and its potential to promote physical fitness is admirable. But it has a number of draw backs. First, prolonged interactivity is fatiguing. Second, it’s designed to detect the movement of large limbs making interaction very cumbersome. Finally, it is not very good at recognizing complex or detailed movements. For these reasons creating an interface that requires full body movements is not ideal. Instead kinesthetic interfaces should focus on small simple actions such as those done by the most common human body part used in performing tasks the human hand. The hand and its appendages have the same relative location and orientation regardless of the situation. But the *Kinect* is not ideal for detecting hand motion in mobile environments because the cameras are stationary and hence made for static environments. It is possible to bridge the gap between kinesthetic interaction and mobile computing through wearable computing. Wearable computing has yet to have the same market appeal as seen in mobile computing, presumably due to technological limitations. It can be expected that as the technology improves, wearable computing will become more accessible, less intrusive, as well as less cumbersome. Current and future mobile interactions will need to be integrated and in some cases redesigned to take advantage of these improvements.

While anticipating these developments, it is important to prototype different techniques that can determine what is and is not effective as the technology becomes more readily available. Text entry is historically the most common way to interface with a machine. In particular, we will focus on mobile text entry. Current trends in mobile technologies show that users are as inclined to text each other as to call and talk [[9](#_ENREF_9)]. As mobile technologies move into the realm of wearable computing, texting will have to migrate as well.

### 2 Problem to be studied

The purpose of the thesis is to assess kinesthetic finger pinching performance and how it is affected by visual/audio/haptic feedback when used in a text entry setting. A proper analysis of hand performance will allow future design interfaces to be built based on the performance and constraints of the hand. The goals are to:

* To build a one handed, minimally restrictive, untethered glove device that utilizes buttons and contact sensors with audio/haptic feedback in an *H4-Writer* algorithm text entry setting.
* To conduct studies with the glove device in order to assess the natural and mechanical constraints of pinching performance of the human hand between the fingers and the thumb.
* Compare difference in fingers and thumb pinching when actuating buttons versus contact sensors.
* To compare performance of haptic and audio feedback for text entry.
* To assess the application and convenience of using such a device in a real world environment when visual feedback is disabled.

Text entry is an ideal performance analysis setting since it is the one of the most common ways to interface with a machine and there are many bodies of work that share standardized measures that can be used for comparison analysis.

### 3 Previous Work

#### 3.1 Overview

Research, on text entry techniques, dates back several decades. Text entry techniques are the way in which a user interacts with a system in order to transcribe text. Techniques may vary for instance the standard text entry technique is the QWERTY technique. The QWERTY technique has a one to one mapping between the keyboard button and its corresponding ASCII character. On the other hand techniques such as the multi-tap use an encoding for every character and as such require several key inputs called codes that enumerate characters [[12](#_ENREF_12)]. The physical implementation of one text entry technique may vary as well. Take for instance *Unistroke*, a text input technique that requires the user to make a distinct stroke gesture in the unique shape of the letter they wish to input [[6](#_ENREF_6)]. This technique has been physically implemented several ways, with a stylus and touch surface, with a mouse, and even by gesturing with your fingers in thin air. In most cases both the technique and the way it is implemented matter in performance. Regardless of technique or their implementations they all share the same evaluation which depends on text entry rate in words per minute, error rate as a percentage, and efficiency as a percentage [[12-14](#_ENREF_12), [18](#_ENREF_18)]. These performance measures are usually captured during transcription which is the process of copying the characters of a predefined phrase. The standard calculation for text entry is the length of a string divided by the average length of an English word (5 characters including the space character) divided by the length of time required to enter that word in minutes [[14](#_ENREF_14)]. The error rate is the minimum string distance (MSD or Levenshtien Distance) over the length of the transcription or phrase whichever is longer [[18](#_ENREF_18)]. The Levenshtien distance is the number of changes operation required to convert one string into another string. There are three change operations which include: insertion, deletion, and substitution. When a transcription and a phrase are the same the MSD is 0 because the strings are identical no operations are needed to convert the strings. Notice that in the case of the QWERTY keyboard the number of keystrokes per character (KSPC) is 1 but on a telephone keypad where more keystrokes are required to enter a single character the KSPC is 2.03 [[12](#_ENREF_12)]. The efficiency is ratio between the number of extra keystrokes per character that were performed over the minimum number of keystrokes required to perform the transcription. This measure captures situation where during transcription a user performs a few mistakes but corrects them reflecting in a 0% error rate but a less than 100% efficiency rate.

Most text entry techniques rely on stationary devices, such as a Qwerty keyboard, or hand held devices such as the *Twiddler* chord keyboard [[10](#_ENREF_10), [11](#_ENREF_11)] or a mobile phone. These devices, although effective, put constraints on the user. For instance, a Qwerty keyboard requires the user to sit or stand in a fixed position and use both hands to input text. Hand held devices are more mobile, but fast and accurate entry usually requires the user to look at the device and use both hands. *Twiddler*, a hand-held device with buttons in a 4 × 3 grid, remedies this by having users press chords to enter text. The device is held in the palm of the user’s hand with the keypad facing away from the user. Lyons et al. report relatively high entry speeds: 27 wpm after 15 hours of practice [[11](#_ENREF_11)]. One user reached 67 wpm after using the device for 10 years. Despite affording relatively high rates, specialized input devices such as *Twiddler* are held rather than worn. We believe that, for wearable computing, users should interact through natural uni- or bi-manual gestures that are relatively unencumbered by devices and unmediated by a held device. One can imagine that the *Twiddler* can be modified to be worn as a glove rather then held, but the creation, performance, and analysis of such a *Twiddler* glove is yet to be seen.

Text entry with a glove has been researched before. Notable devices include the Pinch Glove [[2-5](#_ENREF_2)] and the Chording Glove [[1](#_ENREF_1), [8](#_ENREF_8), [17](#_ENREF_17), [19](#_ENREF_19)]. The performance of these can be assessed in terms of *entry speed* (wpm), *error rate* (%) and *efficiency* (*%*) [[12](#_ENREF_12), [18](#_ENREF_18)], although older papers have a tendency to report only wpm.

Pinch gloves produce discrete actions when the thumb touches, or pinches, the pulpar surface of the distal phalange of a finger. Bowmen et al. [[4](#_ENREF_4), [5](#_ENREF_5)] describe a pinch keyboard which simulates a standard Qwerty keyboard in virtual environments. A QWERTY keyboard is known to have three common rows, the top row, the home row and the bottom row. The QWERTY interface and the selected row could be seen at all times through the systems virtual heads up display. To maintain *KSPC* = 1, interaction requires both hands and numerous arm positions and hand pinch motions to map input to the 26 characters of the alphabet. Stretching out the arms away from the body selected the top row whereas contracting arms closer to the body selected the bottom row. It also uses special gestures such as touching two fingers of opposite hands together for functions like Enter and Space. They report entry speeds of 12-15 wpm after one hour of practice for experienced typists [[3](#_ENREF_3)]. The interface and the interaction were designed to look and feel familiar to users of QWERTY keyboards and have the potential to be learned through kinesthesis. Unfortunately the arm movements were too broad and elaborate which resulted in user fatigue. It is important to note that when designing such a system one has to take into account the performance and agility of the human body. The KSPC for a device may be 1 but if the movement time required performing the keystroke is greater than a few milliseconds than speed will be sacrificed over efficiency.

The *Chording Glove* was originally presented by Rosenberg and Slater [[17](#_ENREF_17)]. Their one-handed glove has three parts: four sensors on the tips of the fingers, three shift buttons on the side of the index finger, and eight function keys on the back of the hand. Entry requires a series of chords (combinations of finger presses against the palm and thumb). For CapsLock and NumLock, the user presses the thumb against a button on the index finger. Function keys require the other hand to actuate. Since every character has a unique input, this technique has 1 *KSPC*. They report an entry speed of 16.8 wpm after 10 hours of practice. This is a reasonable text entry speed. This speed could have been even faster. The chords used in text entry were designed such that the shape of the fingers would resemble the shape of the English letter. This kind of design is ideal when the intention is to easily memorize the technique, but it lacks in efficiency. Some characters in the English language are more common than others and in order to improve efficiency and speed while reducing fatigue it can be important that the most common letters have the simplest and quickest chords to perform. Memorization in these tasks is important but as discussed earlier the human kinesthetic sense will be able to adapt to unconventional actions when needed.

Another text input method is Braille-based chording with a glove. These two handed gloves had touch contacts at each phalange of each finger in the hand. In total one hand had 12 touch sensors and the phalange sensors were activated using the thumb [[1](#_ENREF_1), [8](#_ENREF_8), [15](#_ENREF_15)]. Eight sensors each located on the distal phalange represented a Braille dot. The three proximal phalanges closest to the thumb had special character representation such as mode keys, backspace, space, and brail key macros. Since the gloves were developed for Korean, it is difficult to compare devices tailored for English.

Another glove based text input technique is *Airstroke* [[16](#_ENREF_16)] users pinch their fingers and make *Unistroke* gestures to enter text. It also has the option for word completion. In standard *Unistroke* user would use a stylus in order to draw symbols that resemble the English alphabet that could then be interpreted by the system as character entry. Similarly *Airstroke* has users draw *Unistroke* symbols in the air using two pinched fingers as the tip of the brush. User performance for text entry with word completion was 11 wpm and 6.5 wpm without [[16](#_ENREF_16)]. Error rate with word completion was 6.6% and 11.8% without. Like *Unistroke*, the KSPC for Airstroke is 1; each character is represented by a single stroke. Due to its use of large arm gestures *Airstroke*, in the early stages of the study, was found to be fatiguing to participants, frustrating, and error prone. After a bit of practice and taking a bit more relaxed positions participant performance improved and some even claimed to be able to write most letters “subconsciously” (kinesthetically). Like the Microsoft *Kinect* and the Huffman pinch glove the disadvantage of *Airstroke* are the large physical movements required to perform the character strokes. Although KSPC = 1 to perform any action with a human limb naturally take more time and effort.

#### 3.2 Huffman Base 4 and the H4-Writer

In order to test hand performance we need to use a text entry technique that has a balance between respectable text entry rates with low number of keys. To ensure that we maximize the text entry speed for a low number of keys while maintaining full text entry functionality we need to look at the works of Huffman. Huffman developed an algorithm that generates minimum-redundancy codes for symbols based on the frequency of their use as a means for information transfer. A code is sequence comprised of a combination of discrete inputs. The set of inputs comprising a Huffman codes will be determined by the base which is the number of discrete inputs available. The average length of a set of Huffman codes is equivalent to the average KSPC. The most commonly used symbols have the shortest codes whereas the least commonly used symbols have the longest codes. We employ the previous work based on Huffman coding -- the H4-Writer by MacKenzie et al [[13](#_ENREF_13)].

MacKenzie et al. [13] discuss how Huffman codes are for most part generated using base 2. Through some modeling and analysis they attempt to balance the base and the KSPC in order to find an optimal input method. This achieves the goal of allowing the least number of inputs while maintaining a fast text entry rate respectively. Their analysis found that the optimal base was 4. *H4‑Writer* [[13](#_ENREF_13)] is an optimized four button keyboard that uses base-4 Huffman encoding. It utilizes minimized key sequences to enter letters with full access to error correction, punctuation, digits, and modes, as seen on a standard Qwerty keyboard because Huffman codes are minimally redundant many of them have common stems. This allows Huffman codes to be branched into Huffman trees. The characters and commands were branched into a tree having out-degree 4 at each node. *H4‑Writer’*s *KSPC* = 2.321. In a longitudinal study, users interacted with a graphical interface that mirrored a 4-button gamepad. By pressing buttons on a real gamepad with the thumb, the user branches down the code tree until a leaf node is reached. As buttons are pressed, the graphical on-screen interface updates the display with the remaining characters in the current branch. In the *H4‑Writer* evaluation [[13](#_ENREF_13)], participants transcribed lowercase phrases with no punctuation for 10 sessions to facilitate learning. An 11th session tested comprehensive performance with phrases containing uppercase characters and punctuation. Users averaged 20.4 wpm on their 10th session and 16.5 wpm on the 11th session. Overall, mean error rates were <1%. These results convincingly demonstrated that relatively high rates of accurate text entry are attainable with an easy-to-learn technique that requires only four buttons.

### 4 System Design

A preliminary system was previously built named the *Huffman Base-4 Text Entry Glove* (*H4-TEG*) based on the *H4-Writer*. Its purpose was to gather data and assess system design on finger pinching performance. A mathematical model was created for the *H4-TEG* that was based on a paper by Cole and Abs [[7](#_ENREF_7)]. It reported the average pinching speed between the thumb and index finger was about 280ms. The pinching speed was generalized and applied to all other fingers. The calculations were used as a bench mark that could be compared against study results.

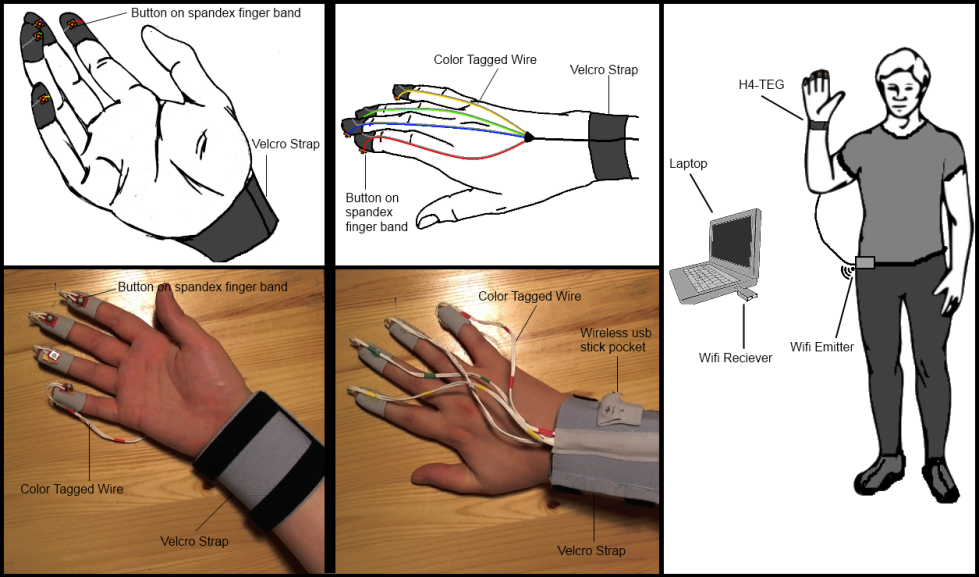
To calculate the benchmark the pinching estimate was multiplied by the length of the *H4-Writer* code for each symbol (1).

|  |  |  |
| --- | --- | --- |
|  |  | (1) |
|  | - |*Hc*| = the length of the H4-Writer Code for symbol c  *- tc* = the estimated pinching time to enter symbol c |  |
|  |  |

Next those symbol time estimates were ran through the text entry rate equation over a set of standard English phrases (2). The results of the model estimated a promising text entry rate of 18.1 wpm.

|  |  |  |
| --- | --- | --- |
|  |  | (2) |
|  | *- phrases* = the phrase set  *- p* = a phrase from the phrase set  *- c* = character from the phrase set  *- tc* = the character entry time (ms) from eq. 1  - “*# of phrases*” = the number of phrases in the set |  |

The glove was built with the intention to minimize hindrance to performance. The overall emphasis was that the glove was to be unrestrictive and non-cumbersome. The general design can be described as a glove nervous system (**Figure 1**).



**Figure 1. Design sketches and components of *H4-TEG***

Spandex sleeves were used to hold the switches in place on each fingertip. Wiring ran over the finger and down the back of the hand to a *Velcro* wrist strap that was used to bundle the wires. The switches on the finger tips were 3 mm in diameter and required 1 newton of force to actuate.

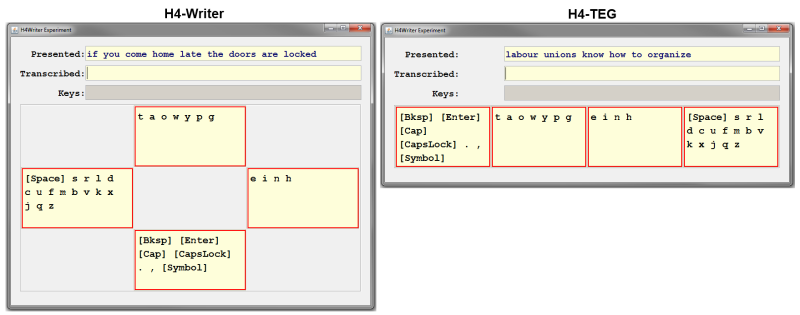
The system communications hardware was reengineered from a Microsoft *Wireless Keyboard Series 800* (**Figure 2**). Certain keys on the keyboard shared common circuit board pins. In this case the keys 0, 1, 2, and 3 on the keyboard shared the same pin. Each respective pin was soldered to the wires of each finger switch. The device was powered by two AAA batteries.



**Figure 2. Microsoft *Wireless Keyboard 800* and the keyboard-processing unit.**

The major perk of reengineering this keyboard was that the glove device could be untethered so users would not have the restriction of being very close to a machine in order to operate the glove.

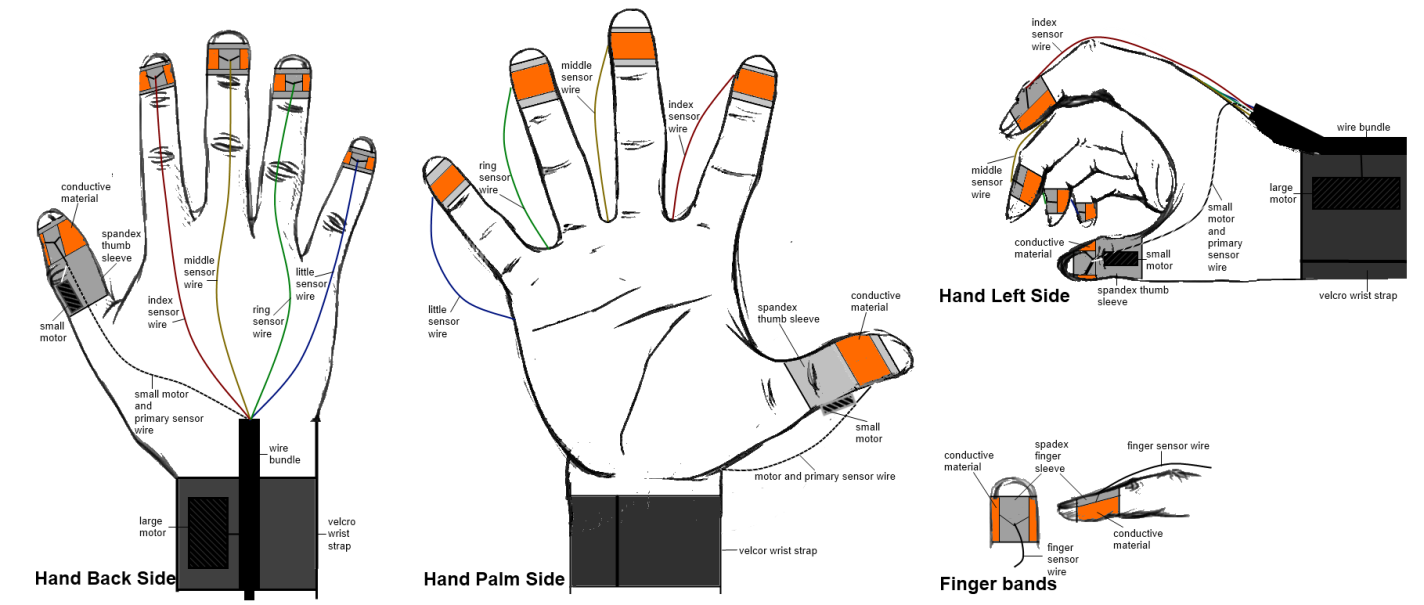
The original *H4-Writer* software interface was modified to reflect the arrangement of the fingers of the human hand (**Figure 3**). This was done so users could map their fingers to the interface while they learned the *H4-Writer* codes. The software also had audio feedback during entry. While a code was being entered a click sound could be heard at every press and a ding sound was heard when a code was completed.



**Figure 3. H4-Writer Layout compared to H4-TEG**

The results of a longitudinal study for the preliminary *H4-TEG* showed error rate was on average 0.34% and an average of one extra keystroke was necessary for every 3.89 keystrokes to correct errors. These were respectable results. Average entry speed was reported to be 14.0 wpm (SD = 3.1) 22.6% below the estimated 18.1 wpm benchmark. The contrast between the reported text entry rate, and the mathematical model highlighted area where the glove design could be improved.

An improved version of the glove is in progress (**Figure 4**). It removes switches from the finger tips entirely and instead uses a conductive fabric to send a signal to indicate a pinching motion. An additional thumb sleeve will be needed in this design in order to be able to complete a circuit when a pinching motion is performed. The conductive material *Soft and Safe tm Shielding* Fabric will be used as the conductive material; it is made of 70% bamboo and 30% silver which makes it comfortable against skin and has a high conductivity (<1 ohm per square). The spandex finger bands have not been changed from the original design.



**Figure 4. *H4-TEG* new design**

The preliminary study also gathered data on relative finger pinching performance between fingers. This data can potentially be used to further optimize *H4-TEG* algorithm on finger performance. The best performing fingers can be used more often for higher frequency symbols much in the same way that those symbols have the shortest codes.

Furthermore in order to expand the gloves features to haptic performance we will add two small motors to the glove. One of the motors will be located on the back of the thumb and the other on the back of the wrist strap. The intention is to remove audio feedback and replace it with light vibrations. When a user is in the middle of a code they feel it on their thumb and when the user finishes a code they feel it on the wrist. Haptic feedback will require the signals to be sent from the glove to the computer and the computer back to the glove. The reengineered Microsoft *Wireless Keyboard Series 800* can no longer used for the study; its wireless communication is unidirectional from the glove to the computer. To accommodate this feature a Microsoft *Xbox 360 Wireless Controller* will be reengineered. The controller is a bi-directional 2.4 GHz wireless device that allows for 9 meters of range. The controller already has built in adjustable vibration feedback and runs on AAA batteries.

The communication with Microsoft *Xbox 360 Controller* is not native to Java. In order to achieve this we will use a readily available Java Xbox 360 library available online and developed by Dr. Aegidius Plüss professor at the University of Berne (Switzerland). It exposes all the features of the wireless controller including the vibration feedback which will allow it to be integrated into the H4-Writer software.

### 5 Goals of the Study

The goal is to develop a physically unrestrictive, untethered glove device that utilizes the H4-Writer algorithm for text entry with audio/haptic feedback. Such a device will allow us to conduct studies that will compare human finger pinching performance between buttons and contact sensors, the feasibility of haptic and audio for text entry feedback. This will also allow us to assess natural and mechanical constraints on user finger pinching performance. With this information we may be able to assess the application and convenience of using such a device in a real world eyes free environment.

Upon completion of the new glove development a longitudinal study will be conducted. The study will consist of two phases: The ten session learning phase where participants will use sensors, audio feedback, haptic feedback and visual feedback, and the two session comparison phase where participants will have different feedback features turned on and off (**Table 1**).

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Phase 1** | **Phase 2** | | | |
| **Session 1-10** | **Session 11** | | **Session 12** | |
| Audio Feedback & Haptic Feedback & Visual feedback | 1/2 Session | 1/2 Session | 1/2 Session | 1/2 Session |
| Audio Feedback | Haptic Feedback | Audio Feedback | Haptic Feedback |
| Visual Feedback | Visual Feedback | No Visual Feedback | No Visual Feedback |

**Table 1. Design of Study**

### 6 Timetable

The glove is already in development and its expected completion date should be mid-July. A study will be conducted near the end of July leading into August. It is desirable to have a thesis completed in time to be eligible for fall graduation 2012.

### 7 References

[1] An, S., Jeon, J., Lee, S., Choi, H., and Choi, H.-G., A pair of wireless braille-based chording gloves, in *Computers Helping People with Special Needs*, vol. 3118). Berlin: Springer, 2004, 490-497.

[2] Bowman, D., Wingrave, C., Campbell, J., and Ly, V., *Using Pinch Gloves™ for both natural and abstract interaction techniques in virtual environments*, Computer Science, Virginia Tech, Blacksburg TR-01-23, 2002.

[3] Bowman, D. A., Ly, V. Q., and Campbell, J. M., *Pinch Keyboard: Natural Text Input for Immersive Virtual Environments*, Computer Science, Virginia Tech, Blacksburg TR-01-15, 2001.

[4] Bowman, D. A., Rhoton, C. J., and Pinho, M. S., Text input techniques for immersive virtual environments: An empirical comparison, *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, Santa Monica, CA: HFES, 2002, 2154-2158.

[5] Bowman, D. A., Wingrave, C. A., Campbell, J. M., Ly, V. Q., and Rhoton, C. J., Novel uses of Pinch Gloves™ for virtual environment interaction techniques, *Virtual Reality*, *6*, 2002, 122-129.

[6] Castellucci, S. J. and MacKenzie, I. S., Graffiti vs. Unistrokes: An empirical comparison, *Proceedings of the ACM Conference on Human Factors in Computing Systems – CHI 2008*, New York: ACM, 2008, 305-308.

[7] Cole, K. J. and Abbs, J. H., Coordination of three-joint digit movements for rapid finger-thumb grasp, *Journal of Neurophysiology*, *55*, 1986, 1407-1423.

[8] Lee, S., Hong, S. H., and Jeon, J. W., Designing a universal keyboard using chording gloves, *Proceedings of the 2003 Conference on Universal Usability - CUU 2003*, New York: ACM, 2003, 142-147.

[9] Lenhart, A., *Teens, Cell Phones and Texting*, Washington, DC: Pew Internet & American Life Project, 2010.

[10] Lyons, K., Plaisted, D., and Starner, T., Expert chording text entry on the Twiddler one-handed keyboard, *Eighth International Symposium on Wearable Computers - ISWC 2004.*, New York: IEEE, 2004, 94-101.

[11] Lyons, K., Starner, T., Plaisted, D., Fusia, J., Lyons, A., Drew, A., and Looney, E. W., Twiddler typing: One-handed chording text entry for mobile phones, *Proceedings of the ACM SIGCHI Conference on Human Factors in Computing Systems - CHI 2004*, New York: ACM, 2004, 671-678.

[12] MacKenzie, I. S., KSPC (keystrokes per character) as a characteristic of text entry techniques, *Proceedings of the Fourth International Symposium on Human Computer Interaction with Mobile Devices - MobileHCI 2002*, Berlin: Springer, 2002, 195-210.

[13] MacKenzie, I. S., Soukoreff, R. W., and Helga, J., 1 thumb, 4 buttons, 20 words per minute: Design and evaluation of H4-Writer, *Proceedings of the 24th ACM Symposium on User Interface Software and Technology - UIST 2011*, New York: ACM, 2011, 471-480.

[14] MacKenzie, I. S. and Tanaka-Ishii, K., Measures of Text Entry Performance, in *Text Entry Systems: Mobility, Accessibility, Universality*, (I. S. MacKenzie and K. Tanaka-Ishii, Eds.). San Francisco: Morgan Kaufmann, 2007, 47-74.

[15] Myung-Chul, C., Kwang-Hyun, P., Soon-Hyuk, H., Jae Wook, J., Sung Il, L., Hyuckyeol, C., and Hoo-Gon, C., A pair of Braille-based chord gloves, *Sixth International Symposium on Wearable Computers Proceedings - ISWC 2002*, New York: IEEE, 2002, 154-155.

[16] Ni, T., Bowman, D., and North, C., AirStroke: Bringing Unistroke Text Entry to Freehand Gesture Interfaces, *Proceedings of the 2011 Annual Conference on Human Factors in Computing Systems*, New York: ACM, 2011, 2473-2476.

[17] Rosenberg, R. and Slater, M., The chording glove: A glove-based text input device, *IEEE Transactions on Systems, Man, and Cybernetics*, *29*, 1999, 186-191.

[18] Soukoreff, R. W. and MacKenzie, I. S., Metrics for text entry research: An evaluation of MSD and KSPC, and a new unified error metric, *Proceedings of the ACM SIGCHI Conference on Human Factors in Computing Systems – CHI 2003*, New York: ACM, 2003, 113-120.

[19] Witt, H. and Janssen, T., Comparing two methods for gesture based short text input using chording, *Extended Abstracts of the ACM Conference on Human Factors in Computing Systems – CHI 2007*, New York: ACM, 2007, 2759-2764.