Facilitating Keyboard Use While Wearing a Head-Mounted Display

by

Keenan Gray

Submitted to the Faculty

of the

WORCESTER POLYTECHNIC INSTITUTE

in partial fulfillment of the requirements for the

Degree of Master of Science

in

Interactive Media and Game Development

APPROVED:

Jeffrey Kesselman Primary Advisor, ________________________________

Jennifer deWinter________________________________

Lane Harrison ________________________________

Abstract

Virtual reality (VR) headsets are becoming more common and will require evolving input mechanisms to support a growing range of applications. Because VR devices require users to wear head-mounted displays, there are accomodations that must be made in order to support specific input devices. One such device, a keyboard, serves as a useful tool for text entry. Many users will require assistance towards using a keyboard when wearing a head-mounted display. Developers have explored new mechanisms to overcome the challenges of text-entry for virtual reality. Several games have toyed with the idea of using motion controllers to provide a text entry mechanism, however few investigations have made on how to assist users in using a physical keyboard while wearing a head-mounted display.

As an alternative to controller based text input, I propose that a software tool could facilitate the use of a physical keyboard in virtual reality. Using computer vision, a user's hands could be projected into the virtual world. With the ability to see the location of their hands relative to the keyboard, users will be able to type despite the obstruction caused by the head-mounted display (HMD). The viability of this approach was tested and the tool released as a plugin for the Unity development platform. The potential uses for the plugin go beyond text entry, and the project can be expanded to include many physical input devices.

Acknowledgements

Many thanks to my advisor, Jeffrey Kesselman who inspired this project. His continual support throughout the project was integral to this project's completion. Jeff's encouragement and technical expertise was vital to the development of the application.

In addition to my advisor, I would like to thank the other members of my thesis committee: Professors Jennifer DeWinter and Lane Harrison. Their feedback and advice enriched this project tremondously. I am especially grateful to Charles Rich [1952-2018] who provided me with the guidance to pursue my Master's Degree.

Thanks to the other Master's students, specifically Mitchell Stevens and Bolin Zhu for help, inspiration, and companionship throughout this process. Last but not the least, I would like to thank my family: my parents Dr. Nancy Gray and Michael Gray, who have never let me down.

Contents

List of Figures

1 Introduction to Virtual Reality Interfaces

Virtual reality (VR) technology presents new problems concerning keyboard use and text entry. Virtual reality devices currently make use of a head-mounted display (HMD). Head-mounted displays are computer displays built into headsets. While there are some “see-through” HMDs, HMDs used for virtual reality tend to be opaque, such that users only see the contents of the display. The inablity for users to see their surroundings poses challenges to the use of destop computer input devices, such as keyboards. Desktop computing devices provide users with responsive, familiar feedback. Keyboards can be designed with deliberate goals in mind, for example keyboards designed for gaming and keyboards designed for high-speed text entry. The design of keyboards is something that can be improved upon along with advances in VR technology to create new novel interfaces for computing. Outside of gaming, computational tasks often utilize the keyboard in some way. Newer apps, such as *Virtual Desktop* (created by Oculus), allow users to use the HMD like a standard display, performing general computing tasks such as emailing or word processing. For more traditional desktop applications ported to virtual reality, a keyboard remains a high-utility device since not all computing tasks will be easily done using motion-based controls.

Many computer users require some degree of vision of the keyboard [6, 16, 14] in order to properly make use of the device. The visual obstruction caused by an HMD can interfere with a user’s ability to use the keyboard and a user’s typing speed and accuracy could decrease as a result [14]. Facilitating keyboard use will help to prevent user frustration when engaged with virtual reality applications that make use of keyboards.

Virtual reality applications make use of gaze for input. The rotation of the head is tracked by the headset and allows for a direct mapping between the actions of the user and the view into the digital world. There are two subcategories of virtual reality games with unique input requirements: room-scale VR and seated VR. In Room-Scale experiences, users are able to use their entire body as an input mechanism. Holding a tracked-controller in each hand, users have the freedom to move about a sectioned off space in the real world - the experience encompassing them completely. In Figure 1.1, the kinetic motion involved in room-scale VR applications is advertised.

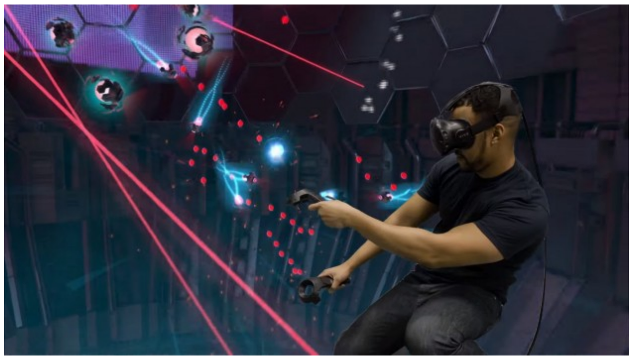


Figure 1.1: *Room Scale VR involves more bodily movement than seated VR, as shown in this advertisement for the HTC Vive [29]*

Seated VR games, as the name implies, involve sitting in a chair and wearing an HMD. Seated VR is well suited for driving or flight simulation games but has also been used for a wide variety of games. Seated VR games are popular for users that do not have the space available for a room-scale system. Games made for the Oculus Rift and HTC Vive typically support both room scale or seated environments. The games themselves make adjustments based on space limitations to accomodate these differences. While seated VR games can make use of tracked controllers in each hand, seated experiences can also be integrated with desktop input devices such as gamepad controllers, keyboard and mouse, or HOTAS (Hands On Throttle-and-Stick).

Both kinds of VR experiences have been rated highly as feeling “realer” to users than traditional desktop games [15]. The differences between both categories of VR applications lies primarily in input design. Room-scale VR requires a prepared space with no detritus or obstructions. Instead of the classical desktop paradigm, input devices for room-scale VR are often held or worn by the player at all times. Figure 1.1 shows an advertisement for the HTC Vive. In the image, a player holds a wireless control in each hand while wearing the headset. The freedom of movement afforded by room-scale technology diminishes the utility of a traditional keyboard. Because seated VR applications are played in a stationary position, there is the potential to integrate the keyboard as an input device for these kinds of games and applications.

How can keyboard use be facilitated despite the challenges presented by an HMD? I propose a solution to the problem using a web-camera pointed at the keyboard. Using computer vision, a projection of the user's hands can be created and displayed over a virtual keyboard. The visual guide will help users employ the hunt-and-peck strategy, and allow touch-typists to reposition their fingers if necessary. The position of the user, camera, and keyboard is demonstrated in Figure 1.2. To test this, a physical keyboard and a web-camera are required and both devices will

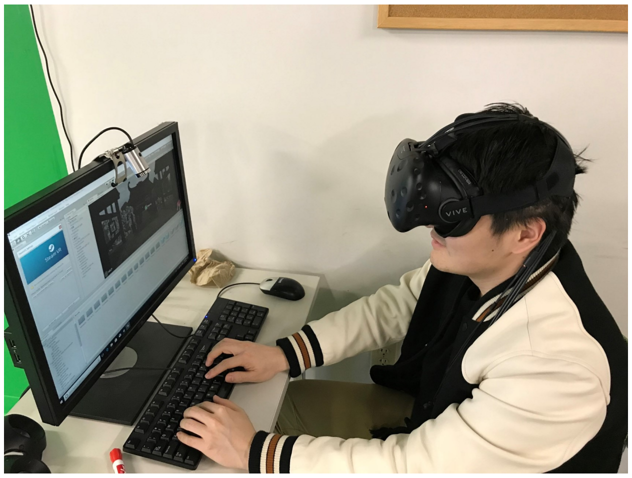


Figure 1.2: *A user sits wearing an HMD. The camera on the monitor will facilitate his use of the keyboard*

remain stationary during use. Due to the restriction the devices, the camera based tracking solution is targeted at stationary VR experiences only. Alternative text-entry devices have been proposed for room-scale VR. One proposal uses a virtual keyboard model that is struck using the motion controllers, as though each key was a drumhead. An assessment of the drum keyboard typing approach will aid developers in designing VR experiences (seated or otherwise) with text-based chat in mind. This thesis intends to answer the following research questions:

1. Can markerless hand tracking provide enough visual information to aid users in keyboard use?
2. How does typing on a keyboard differ from using a controller interface for typing?

Virtual reality devices, such as the Oculus Rift and the HTC Vive, attempt to provide a more immersive user experience, making them a good fit for gaming applications. The popularity of these devices stems from their ability to augment important aspects of gameplay and user experience, including flow, immersion, and presence. Flow, immersion, and presence are game concepts that are positively correlated with user enjoyment and percieved quality of a game [31, 27]. A 2006 study recognized immersion as one of five factors of user motivation in gaming [34]. These factors combined with ever improving virtual reality technology have likely increased the incentives for developers to create virtual reality games. In fact, the Game Developers Conference 2016 State of the Industry Report [2] recognized that the number of developers planning to release games for virtual reality doubled from 2015 to 2016. The tools developers have access to will help to improve the variety and quality of many virtual reality games.

While many VR games do not require a keyboard at all, some games and applications developed for virtual reality will require use of a keyboard for a variety of reasons. One use case serves as the primary focus of this project: text chat. Text based chat is a popular form of computer-mediated communication (CMC) in online multiplayer games. There are already hundreds of virtual reality games on Steam, many of which are also tagged as multiplayer games (SteamSpy.com). Games that offer communication tools for players such as text or voice chat may encounter new challenges posed by virtual reality development. Facilitating keyboard use in virtual reality therefore becomes a challenge worth overcoming.

Text-based chat is an important tool for synchronous *and* asynchronous communication, wherease voice chat is less suited to asynchronous communication. Online games have, for a long time, used text as a communication mechanism between players [11, 19]. In games, chat has been used for both goal-based and social communication [11]. While voice chat is becoming increasingly popular in video games [30], there are still many reasons players can benefit from text chat. Studies of online games, for example, have identified several utilities of text-based chat that make text preferable to voice-based communication. One such benefit is the ability to hear other audio sources (e.g. game audio) while still being able to communicate [8]. Additionally, voice chat can be prone to annoyances, for example, background noise, crosstalk, or griefing. Subjects in another study also found it more difficult to identify speakers in relation to in-game avatars when using voice chat [30].

Users can also have need for a physical keyboard in games where text chat is not a priority. In many games, it can be useful to be able to find specific keys on a keyboard in order to perform certain inputs. The ability to position your fingers visually on the keyboard could be useful for VR games with more complex control schemes. The game *Elite Dangerous* is a popular sci-fi flight simulator that has virtual reality support. While playing the games, players might need to press specific keys on the keyboard in order to activate game-related functions, such as deploying landing gear or adjusting their speed. Outside of virtual reality, users must reposition their fingers either tactiley or visually in order to press the proper keys.

It is likely that as human-computer interactions evolve, head-mounted displays may replace traditional desktop monitors as a comfortable display device for work, travel, and leisure. Understanding the current limitations of the technology and efforts towards a solutions will contribute to the advancement and adoption of virtual reality technology.

The specific contributions of this report are

1. A low cost solution to the impairment of keyboard use caused by a head-mounted display.
2. Results from a study on the effectiveness of a web camera utility for helping users type while wearing a headset.
3. Results from a study on the effectiveness of a “Drum Keyboard” for text-entry in room-scale VR.
4. Feedback from users on the viability of each approach as it relates to certain typing tasks and user skill gaps.

The sections of this paper are organized as follows: In the background chapter I provide information on the current understanding of how people type on physical keyboards as well as reviews of other typing studies specific to virtual reality text input. In the methodology, I explain the design goals and programming methods used to achieve these goals. An in depth explanation of the experimental methods used to assess the software in this thesis is also provided. In the results chapter, data proving that users prefer having a visual aide when typing are presented. Additional results, such as the efficiency of a more kinetic typing strategy for virtual reality are presented. In the conclusions chapter I will present the improvements made to the tool post-assessment, and present possibilities for future work surrounding both the specific utility and physical interfaces for VR.

Ultimately this work will explore the possibility for physical interfaces to remain relevant for use with head-mounted displays. Future applications that make use of virtual reality can benefit from interfaces that have been adapted for use with VR. Finally, this work explores computer vision to functionally overcome several challenges presented by virtual reality head-mounted displays, which isolate users from their real world surroundings.

2 Background: A Review of Typing in Virtual Reality

In order to have the necessary background needed to conduct this work I examine the utility of computer keyboards and the affordances that facilitate effective text-entry. I present a closer look at answering the following question related to virtual reality input devices. What input mechanisms have been studied for use with VR technology?

Throughout the development of commercial virtual reality hardware, interface challenges have been identified, and have inspired many studies and projects, including this thesis. To evolve and grow as a technology, virtual reality games may require the adaptation of input devices such as the keyboard. Adaptation of traditional input devices affords designers the opportunity to design a broad scope of interactions.

Virtual reality is rapidly growing as a technology of interest and integration with well-understood interfaces is important. For example, advances in speech-to-text software allow for fast and clear text input using speech [30, 11, 8]. Additionally, voice-chat is becoming more efficent and is commonplace in online video games. Because of trade-offs between voice and text-based chat, keyboards remain an important tool for communication in games. Keyboards should still be considered a viable input device for virtual reality since they are commonplace.

2.1 How People Type

When using a physical keyboard for text-entry, there are two commonly agreed upon strategies: Touch typing and hunt-and-peck. The simple definition of touch typing is a user that can correctly type using only their sense of touch. Touch typing is more rigidly viewed as the explicitly taught method through which all fingers are used to type. Touch typists do not require sight of the keyboard in order to accurately position their fingers or in order to find the intended keys. Touch typists typically recalibrate their finger position using the nubs on the “j” and “k” keys. The hunt-and-peck method typically involves fewer fingers; at least two, but sometimes more. The hunt-and-peck method has traditionally been thought of as being slower and requiring a view of the keyboard [5]. However these views are being refuted by more modern studies. Despite the fact that non-touch typists (called “everyday typists” by Feit) are not formally trained, they can achieve comparable speeds as trained touch-typists [6]. Everyday users generally require a view of the keyboard in order ensure that they are entering the correct keys [6, 16].

Because everyday typists are not formally taught, there tends to be a variety of different strategies employed when typing. For example, the fingers used to hit certain keys will vary from user to user [6, 17]. The high degree of individual variance [17, 26] in how people type makes it difficult to determine which factors affect a typist’s ability. Of particular interest to this study is vision of the keyboard. Occlusion of the keyboard from the user has been shown to descrease typing speed and increase error rate, as has wearing an HMD [14].

In a broad study about typing methodology, Anna Maria Feit and her coworkers presented three relevant findings.

1. The number of fingers used while typing is not a large component of typing speed.

2. Trained typists are not necessarily faster than self-taught typists.

3. Irrespective of typing speed, non-touch typists spent more time looking towards their fingers, and the keyboard, when performing the typing tasks.

The first two points here are corroborated in an older paper that investigated why users trained to touch-type sometimes ameliorated typing effectiveness by reverting to a visually based hunt-and-peck approach [33]. Another paper [16] provided a deeper analysis of users that looked at the keyboard while typing. Instead of separating users into two group (touch-typists and hunt-and-peck typists) this study used eye tracking technology to separate users based on time spent looking at the keyboard. The study grouped users into “keyboard gazers”, “monitor gazers”, or “mixed-strategy writers” depending on gaze behavior. In the experimental trial, the majority of users exibited mixed-strategy or keyboard gazing tendencies. It was rare for users to type while looking at the monitor alone [16].

Feit's, Johansson's and other's investigations into keyboard use behavior are particularly important to this research since the studies note that some form of visual feedback is often necessary for typing tasks. Additionally, these papers provide some information about the possible distribution of touch and non-touch typists. Feit's study had 43% touch-typists, with the rest being classified as non-touch typists. Another study found 57% touch-typists [17] compared with hunt-and-peck typists or mixed strategy users. The sample sizes in these studies were small but suggest that somewhere around 50% of computer users can touch-type. For everyday typists, touch typing is a difficult skill to learn [33, 5] so alternatives should be considered in order to promote wider adoption of virtual reality devices. The statistics on typing strategies provided insight into what form the virtual reality typing mechanism would take. For example, some users would require minimal guidance to use a physical keyboard. Everyday typists would potentially prefer tools that do not require the memorization of a keyboard layout or the repetetive training necessary for touch-typing.

2.2 Interfaces for Text Entry in Virtual Reality

This work does not seek to provide a "best" solution for text-entry in virtual reality. The choice to facilitate use of a physical keyboard, using computer vision, comes from an understanding of additional work done in this field. There is abundant research into overcoming the challenges presented by HMDs. The following section summarizes some of the research in the overlapping areas of virtual reality, text-entry, and human-computer interactions.

2.2.1 Speech-To-Text

A possible text-entry strategy for virtual reality is speech-to-text. There are many useful applications for speech-to-text software, but there is still a case to be made for keyboard-based input in contemporary virtual reality applications. Voice input is often viewed as the ideal mechanism for text entry. A Stanford study compared text-input speeds for voice input and touch-screen keyboards. The study found that voice entry could be much faster than using a touch-screen keyboard [25]. Speech recognition software has advanced considerably in the past few years, allowing for high entry speeds and accuracy. Speech has also been used as a game input device. Experiments with the Xbox Kinect, indicated players were able to successfully provide game input through voice commands. Additionally, in 2014, IBM made a prototype for a game that is played almost entirely using voice commands [4]. The introduction of speech-to-text software did not remove the necessity for a keyboard as a means of text-entry. Text messaging, for example, is a prominent form of computer-mediated-communication that can be done either using a phone's digital keyboard, or through speech-to-text. Familiarity helps to ease into transitis in hardware and software.

Instead of using speech-to-text, direct voice chat is another option for online communication. As mentioned in the introduction, there are tradeoffs between voice and text-chat including the ability to hear game audio [8]. Before online voice-chat became wide-spread, text was the de-facto tool for communicating with other video game users [11, 19]. Players can also benefit from text chat. Wadley and his team found that voice chat can result in unpleasant interactions, such as griefing. Voice chat is also prone to background noise and cross-talk [30]. Additionally, privacy becomes a concern when using speech-to-text software.

Another application is to combine the two approaches. It is possible to use speech-to-text software to take in a users speech, convert this input to text, then broadcast the message to other players. Little (if any) work has been done on synchronous speech-to-text communication within games, but it is not unlike text-messaging using speech to compose messages. Work in this area has primarily focussed on inclusion and accessibilility for players with hearing disablities. Microsoft has implemented a speech-to-text system in several games that converts voice chat to text to make communication between hard of hearing or deaf players possible [28]. Other use cases for speech-to-text have focused on autonomous agents, like chat bots [22] or chat-log transcription [28].

2.2.2 Reducing Typing Errors While Wearing an HMD

Once developers recognized that typing with a head-mounted display may prove difficult to many users, several projects investigated ways to reduce error rate and improve the keyboard interface for VR. A study by Walker et al. utilizes a text decoder to correct user error while typing with an HMD. The decoder study looked at decreasing this error rate through automated error correction for entire sentences. Decoding complete sentences took non-trivial computation time, and as a result there is some delay in when the message is typed and when it is corrected and finally sent. To study the effect of the decoder on user error rate, these researchers analyzed users typing ability across three conditions: visible keyboard, occluded keyboard, and while wearing an HMD.

Walker's work supports the idea that head-mounted displays will disrupt keyboard use. In this work, a similar methodology is used to study the impact of facilitating keyboard use through computer vision. This work differs in its focus on real-time keyboard use as well as text entry. Once the text has been entered by the user it should be possible to autocorrect words similar to software found on many mobile devices. Text-prediction is a valuable tool for error correction and would certainly be useful for VR keyboards [21]. However, text correction is not the focus of the software tool and so is beyond the scope of this project.

Another mechanism for reducing error rate was explored by using a web-camera to open a window within virtual reality to the outside world. In the study, the video feed from a webcamera was mapped to a texture in the virtual world. The camera feed - showing the keyboard, users hands, and desk space - allowed for a reduction in error rate [20] when typing. Users in this experiment were able to quickly identify the position of the keyboard in the real world and begin typing. The study provided a clear starting point for further research. If a complete image of the space surrounding the user aided in typing performance, it seemed possible that abstracting this view, by creating a virtual keyboard model, would still help a user type accurately.

The possibility for abstracting a virtual keyboard has been explored in several ways. Two companion studies provided important considerations which influenced the design of the virtual keyboard for this work. Each study investigated different aspects regarding virtual keyboard representations for use with head-mounted displays.

1. Where the virtual keyboard is positioned relative to the HMD user [10].

2. How to virtually represent the hands [9].

These papers argued that the physical keyboard remains an important tool for virtual reality applications.

2.3 Alternative Methods for Text-Entry

Virtual reality headsets sometimes include specialized input devices. The HTC Vive and the Oculus Rift both have handheld controllers that are tracked relative to the head-mounted display. These devices have already been used in several games requiring text input. To facilitate text input in these games, virtual keyboards have been created. Users then type by using the tracked controllers to hit the keys on the virtual keyboard as though they were playing the drums [32, 10]. An example of a drum keyboard is seen in Figure 2.1 Other forms of the drum approach could be

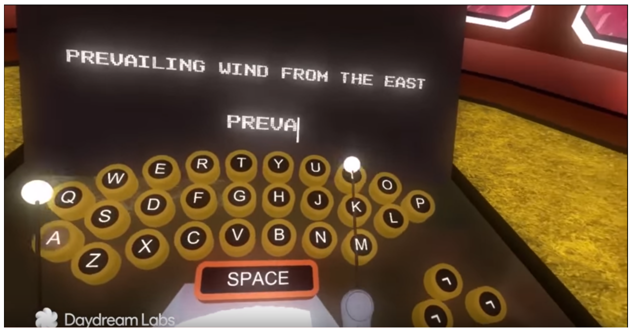


Figure 2.1: *A drum based keyboard utilizes two tracked controllers to hit keys like drumheads*

created using tracked gloves, or specialized cameras like the Leap Motion controller. The Leap Motion Contoller can track the hands of a user directly - turning the users' hands alone into the input device. A drum keyboard that makes use of the Leap Motion Controller is pictured in Figure 2.2. The Leap Motion Controller tracks hand position well, and can be useful for recognizing gestures. Gesture based input presents its own share of challenges including the lack of haptic feedback. Also, the Leap Motion Camera is typically attached to the HMD, meaning that users must face their hands in order to track them properly.

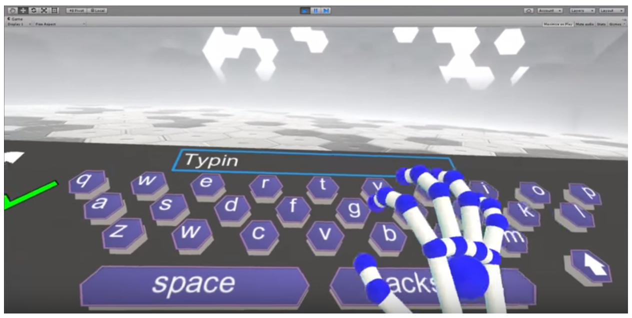


Figure 2.2: *An implementation of a VR keyboard that uses the Leap Motion Controller [3]*

Text input and keyboard use in virtual reality are both important components of HCI research. Alternate typing mechanisms that have been developed for some apps have not been extensively studied. *Editor VR* and *Fantastic Contraption* are two example games that have used drum keyboards. An understanding of the benefits of a drum keyboard would be helpful to developers that plan to include such interfaces in their games. The company, Normal.VR, has created an open-source version of a drum keyboard and encourages developers to test and modify it to suit their needs.

Keyboards that work entirely by tracking the user's fingers are another interesting piece of technology. One such keyboard, the Canesta Keyboard [Figure 2.3], projects an image of a keyboard on a flat surface [24]. Users type by hitting keys on the surface. In the Canesta study, touch-typists felt at a disadvantage since there was nowhere to rest their palms while typing [24]. Future work could study the effectiveness of these devices, which may prove more portable than traditional keyboards. Typing on flat surfaces, such as touchscreen or the Canesta Keyboard has been shown to be slower than typing on a physical keyboard. This may be in part due to many users typing with fewer fingers [10].

The analysis and utility of drum keyboards has not been extensively studied. In this work, the Leap Motion's tracking was found to be too inaccurate for input on a virtual keyboard. However,

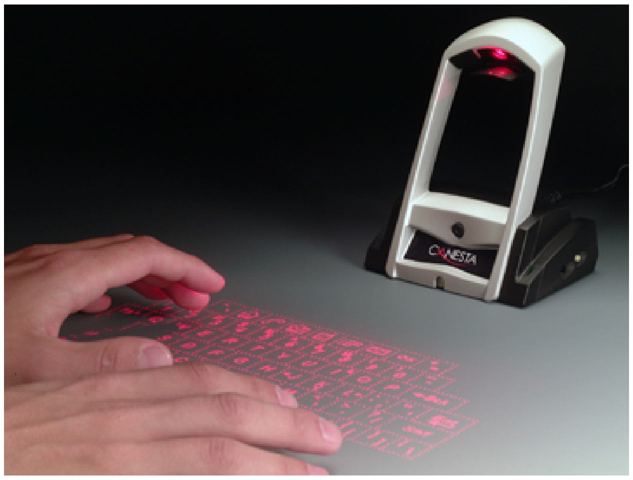


Figure 2.3: *The Canesta keyboard uses hand recognition to allow for typing on any surface [13]*

.

a decision was made to test a drum keyboard as a point of comparison with the physical keyboard. A comparison of these systems is useful to designers since they may want to include a specific text-entry system or chat approach in order to facilitate certain kinds of interactions. As a result, it is important to compare these systems and identify which system is better suited for specific typing tasks.

Another hand tracking solution used a phone capable of finger tracking [18]. The phone is able to track the nearby position of the user's fingers using its "Hover-Touch" capability. Within the HMD, the user is able to see a representation of the phone, with a touch keyboard that correctly displayed which keys the user was hovering over. Similar finger tracking technology could be added to a keyboard in order to make keyboards feasible input devices for virtual reality. Tracking the keyboard position in this way would allow users to use their physical keyboard, despite the head-mounted display. Input devices that use hand tracking, such as the Canesta keyboard and drum keyboards could potentially be useful for augmented reality applications as well. Augmented Reality (AR) devices do not inhibit a user's view of the outside world so the computer vision tool presented here is less applicable to AR. Despite the growth of the augmented reality field, virtual reality still provides immersive experiences that cannot be replicated with other kinds of headsets.

There is documented commercial viability in virtual reality keyboards. In November 2017, Logitech sent out development kits for a virtual reality keyboard that tracked a user's hand. Additionally, the keyboard was wireless and could be moved and positioned in room-scale applications using a tracker on the controller [23]. The Logitech BRIDGE (as the keyboard is called) uses the HTC Vive's tracking to mark user's hand position and draw a 3D hand model in VR. The hand tracking code used in the Logitech BRIDGE is unpublished, and the mechanism for tracking the hand is proprietary information held by the company.

Keyboards have evolved over one-hundred and fifty years and are recognized as a dominant form of text entry. Virtual reality presents a new domain for keyboard use and the device's potential should not be outright dismissed. There are several mechanisms already employed for text-entry in virtual reality but each solution presents certain obstacles. It is clear in this background that keyboard interaction will remain important to users as virtual reality develops. Additionally, comparisons are necessary between standard keyboard use and other virtual keyboard designs for optimizing the user experience.

3 Methods for Creating and Testing the Computer Vision Tool

Before development could begin, a review of the current literature was completed. Observations from the literature review showed that a markerless approach to hand tracking would be of potential use. Other virtual reality studies exhibited repeatable experimental techniques co-opted for this thesis. After gaining an understanding of keyboard dynamics and typing strategies the development of the typing solution began.

3.1 Design Goals

The final application was designed with several goals in mind. The first goal, was to have an optimized finger tracking solution that would allow for users to understand how their fingers are positioned over the physical keyboard. Maximizing performance was the second goal. Maintaining a high frame rate is a requirement for virtual reality applications as a measure to prevent motion sickness [7, 12]. In order to avoid constraining other developers, the proposed solution must have little to no impact on the performance of any VR game. Another goal for the project was to provide a low-cost, minimal setup utility, done without markers (tracking points placed on the hands). Instead, users would only need a webcamera, in essence making the system ”plug and play” for keyboard facilitation. Reducing the barriers to entry for this tool would aid in wider adoption of the tool for virtual reality apps.

Finally it is also important that the tool supports users of varying skin tones as well as different types of webcameras and keyboards. Many computer vision tools have failed in regards to diversity and only work for white-skinned users. In the same vein, hand accessories such as jewelry, nail polish, or tattoos, should not limit a user’s ability to use the tool. Unity is a premiere tool for the development of VR applications, and the solution was designed for simple integration into the engine. As a Unity plugin, the tool is available to all developers. Clear documentation is also provided for the tool. Additional smaller goals included an understandable setup interface, and utilities to modify the virtual keyboard to suit developer needs.

The goals outlined here work to support and fulfill the research questions of this project. In order to answer the research questions presented in the introduction, more specific questions were investigated in an experiment.

1. How much faster can users type when using the visual aid?
2. Do users make fewer errors when given a visual aid?
3. Do users feel comfortable using handheld controllers to type?
4. Are there reasons users might prefer typing on a physical keyboard, even while wearing a HMD?
5. Can users comfortably adapt to input devices that require more bodily movement?
6. Can users type accurately with a controller interface?

3.2 Development of a Unity Plugin for Hand Imaging

3.2.1 Software Dependencies

This project is composed solely of appropriately licensed open-source code and personally created code. The keyboard facilitation tool makes use of Unity, a software platform for game development. A Unity GameObject provides the tool's functionality to any Unity Developer that downloads the plugin. Hand tracking is done using a dynamically linked library (DLL) written in C++. The DLL was created using OpenCV, an open-source computer vision libary. OpenCV supports many platforms, including Windows, Mac, Linux, as well as Android and iOS for mobile devices.

Through the inclusion of managed plugins, Unity's scripting language, C# allows for the calling of external C++ functions and the transmission of data between the game engine and dynamically linked library. A C# script within Unity passes finger detection data between the DLL and the Unity GameObject. Figure 3.1 shows how the physical and digital components fit together. The diagram emphasizes that any developer that wishes to use the tool can include the keyboard Gameobject in their unity scene. The required OpenCV libraries will be included with the Unity plugin making the tool widely available. Installation is possible with as few steps as possible because the components are linearly arranged.

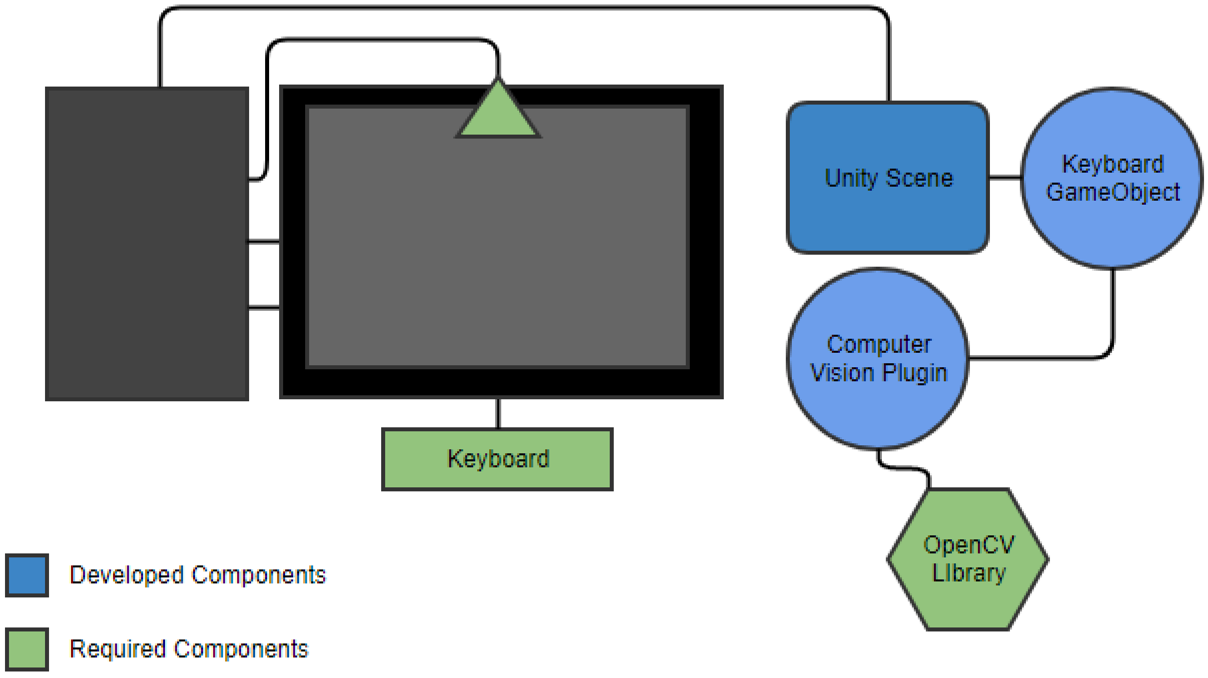


Figure 3.1: *A diagram illustrates the components of the system implemented and how they fit together.*

Once the programming tools were selected, the next steps explored how to make support for many users and keyboard types possible. For a time we considered building a custom keyboard that could track fingers through capacitive sensing. This approach was abandoned in favor of a more universal approach, leading to the decision to use computer vision to track a users hands. Any standard USB webcamera could potentially serve as a facilitation device. Computer vision software offers several advantages including portability and more general support for many keyboards. A webcamera provided a much more affordable solution than other devices like the Leap Motion or Logitech BRIDGE. Developers familiar with virtual reality may already own their own webcameras. Exploring the viablity of a low-cost tracking solution was also a major focus of this thesis.

The next development steps of the project involved tracking a users hands. A standalone C++ project was created could familiarize myself with the OpenCV library as well as computer vision techniques. In the course of this learning phase I followed a guide on passing data from a DLL and Unity GameObjects. After I learned the mechanism for communicating between Unity and OpenCV the next challenge, isolating a user's hands from a video stream, was addressed.

3.2.2 Programming the Tool

A significant amount of time on this project was spent investigating the multiple approaches that can be implemented to isolate, track, and process a user's hands over a keyboard. At first, attempts were made to locate both the keyboard and a user's hands using computer vision. Creating an algorithm to identify the bounds of a keyboard proved difficult and was in fact unnecessary to the functionality of the tool. This approach was abandoned in favor of solely tracking a user's hands. Instead of requiring the webcamera to identify the keyboard outline, users were able to crop the webcamera's image in order to square the keyboard in a frame. Allowing for the user's to manually select the keyboard bounds made the program more general, as the keyboard does not need to remain directly below the camera. Figure 3.2 shows two example configurations where the webcamera is positioned at different distances from the keyboard. Whether or not the camera is distant or proximal, the utility still functions. The hand

Figure 3.2: *The setup for the utility does not require the keyboard to be centered in the frame, or of a specific distance from the camera.*

detection pipeline used to track a users fingers consisted of a sequence of image transformations common in computer vision applications. After opening an image stream from the webcamera a series of steps is followed:

1. Orient the image as desired.

2. Convert image to binary.

3. Isolate the hands.

4. Detect edges of hands.

5. Edge data is stored in a Matrix.

6. Matrix is converted to list of points.

7. List of points is shared with Unity, and rendered.

For hand tracking, the critical step in this pipeline was isolation of the hands from the keyboard. What initially appeared to be an easy problem, removing the static keyboard, and filtering out only the hands from an image proved to be quite difficult I faced several problems including, noise, speed, and sensitivity to lighting changes. Three different approaches were tested to resolve this challenge before settling on the PMOG foreground segregation algorithm [36, 35].

In initial attempts at isolating the hands, image thresholding was used in order to highlight specific regions of color within the image. See Figure 3.3 for an example of image thresholding for the color green. This approach failed on two counts. Primarily, thresholding skin-tones requires a non-trivial amount of fine tuning on an individual basis. Fine-tuning during setup of the application would present a barrier to new users. Second, thresholding the hands did not work unless a high degree of contrast existed between the user's hands and the keyboard color. Reflections from overhead lights on both the keyboard and the user's hands posed an additional challenge to a threshold-only approach.

(a) *The source image*

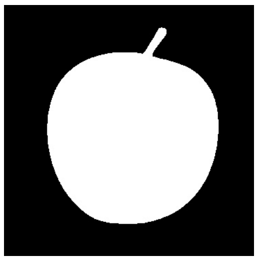

(b) *A mask created by thresholding the image for the color green.*

Figure 3.3: *Example of a thresholding operation performed on an image of an apple.*

The second approach for isolating a users hands tested if it was possible to remove pixels from the image that closely matched an image of the keyboard taken from the webcamera. By subtracting portions of the image that matched the keyboard, only the hands would remain. During the setup of the camera, an image of the keyboard was taken. Each frame of the web-camera's video was then compared, pixel-to-pixel, against the original keyboard image. A series of additional transformations, blurring and thresholding the image, resulted in a well isolated hand [see Figure 3.4].

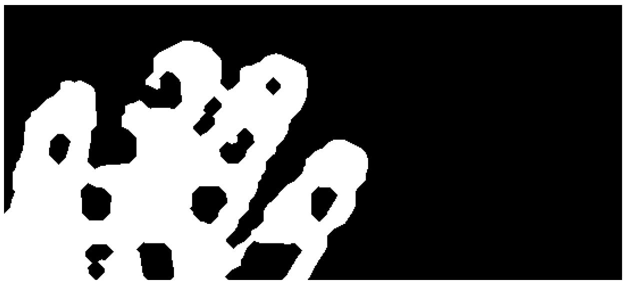


Figure 3.4: *A mask of the hand created by the second isolation technique*

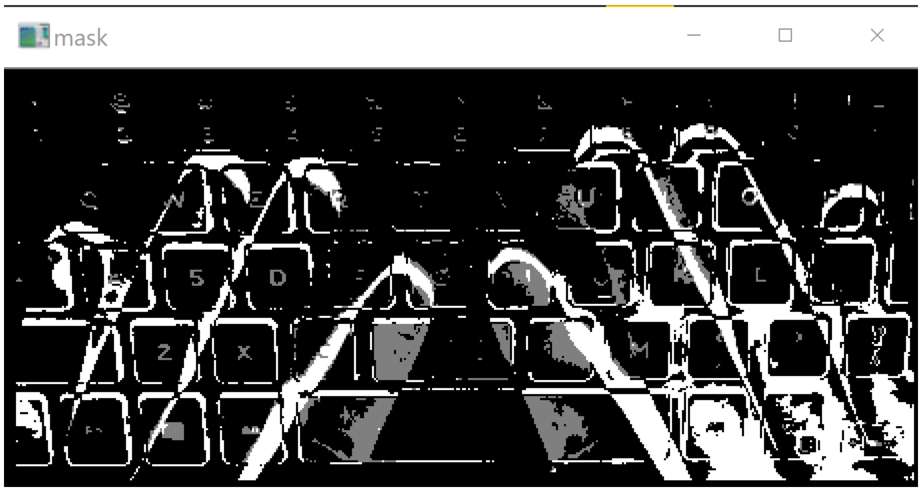
Unfortunately, this approach was sensitive to changes in the lighting and reliance on specific threshold values meant that different skin-tones or keyboard colors were unsupported. Another problem with the second approach was keyboard movement. If the keyboard was moved or bumped accidentally by the user, then the tracking would be disrupted.

More advanced algorithms provided the solution. Another utility provided by the OpenCV library is a background subtraction mechanism called PMOG. Similar to other approaches, the

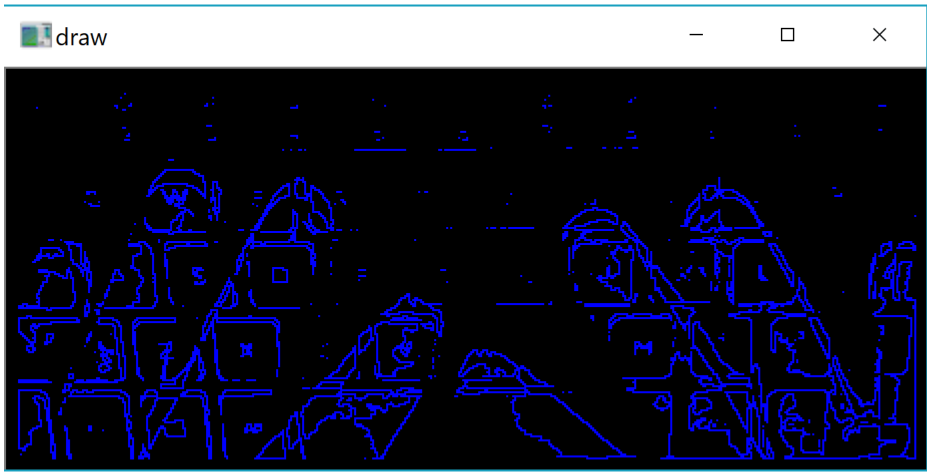
PMOG function attempts to segregate the background and foreground, allowing the hands to be isolated. First, the algorithm captures several frames of video input and establishes an idea of the background. Establishing the background model was similar to taking the initial image of the keyboard in the previous technique. Where this technique differs is in its ability to combine several frames of input into a background model using a Guassian Mixture Model [36, 35].

The background model is used to compare against the current frame of input such that any new visual information, such as a user's hands can be isolated. In the call to the PMOG function, a threshold value is specified, adding some flexibility to the amount of ”new” information that will be selected as part of the foreground. Finally, the PMOG algorithm can be set to update the background model over time. In application here, updating the background model is not necessary since keyboard placement is static. However, the function call in OpenCV can be modified to allow for keyboard repositioning.

Once the hands were isolated, the Canny [1] edge detection algorithm [see Figure 3.5b] was run in order to aggregate position data into a matrix. The FindHands function makes use of the PMOG and Canny algorithms to isolate a user's hand from the static keyboard background. Output from each step of the FindHands algorithm is shown in figure 3.5.



(a) *Output of the PMOG function. White pixels show the foreground mask, pixels in the background were rendered black. Grey pixels represent portions of the image interpreted as shadows*



(b) *The Canny algorithm detects edges in the segmented image*

Figure 3.5: *The steps in the "FindHands" function*

Figure 3.5b shows that some odd edges are detected in places where they are not expected. Most noticibly, some key edges are marked despite being blocked by the hand. Fortunately, it proved easy to establish more deliberate edges by an additional layer of processing. After the edges were detected, the list of points was analyzed and contours of small size are removed. Retaining only the largest contours eliminates the small outlines of the keyboard keys and letters, leaving only the hand outlines.

Figure 3.8b shows the hand outlines as they appear in Unity. The thickness of the outlines was adjustable but the outlines were not filled. A benefit of the transparency of the outlines served as an augmentation of the physical body. Through the projection of a users hands, less of the keyboard was blocked and users could search for the intended keys more easily.

3.2.3 Interactions with Unity

Once the hands could be successfully isolated, the next step of the process involved moving data between the library and Unity. Initially I worked to calculate finger-tip position in the vision application. Because keyboard users employ their finger tips to hit the keys, visualisation of the fingertips alone would allow for keyboard use.

The initial plan was to pass only finger-tip position upwards to the Unity application. Within Unity, spheres (or some other object) could be placed over a virtual keyboard to indicate where the users actual fingers rested above the keybaord. Fingertip representation proved difficult without a marker based approach and while some limited tip detection was achieved it was far too inaccurate. At this point an alpha version of the project was shared with several other students and professors. Feedback from the alpha version seen in Figure 3.6 was valuable to the improvement of the final product. Testers of the alpha found it difficult to identify their exact finger position due to noise. This limited preliminary testing of the application proved the model was insufficient. Additionally, the default spheres used to represent finger tips were unattractive and as a whole, the system lacked sufficient visual information.

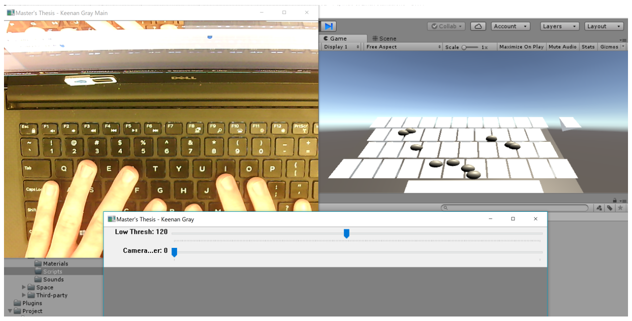


Figure 3.6: *An alpha version of the software shows the less accurate tracking, as well as a seperate interface adjustment and setup*

The next segment of the project was launched with renewed vigor to improve the polish and appearance of the virtual keyboard. A decision was made that the fingertip tracking was too inconsistent to explore further. Instead, the virtual hand would be recreated by redrawing the outlines of the hands in Unity. Unity has a built-in GameObject called a LineRenderer that accepts a series of points and draws lines between the points, creating a polygon. The list of points output by the hand detection functions is unordered, a single line render connecting all of the points did not result in a polygon of the hand, but rather a mess of horizontal lines. In Figure 3.7 are shown large linear segments that spread from points on the hand to the edges of the screen.

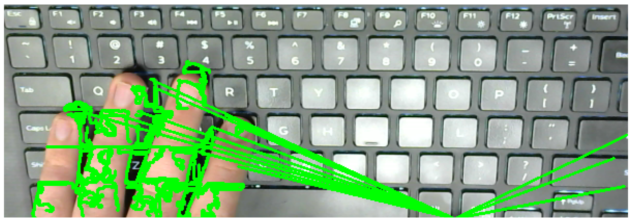
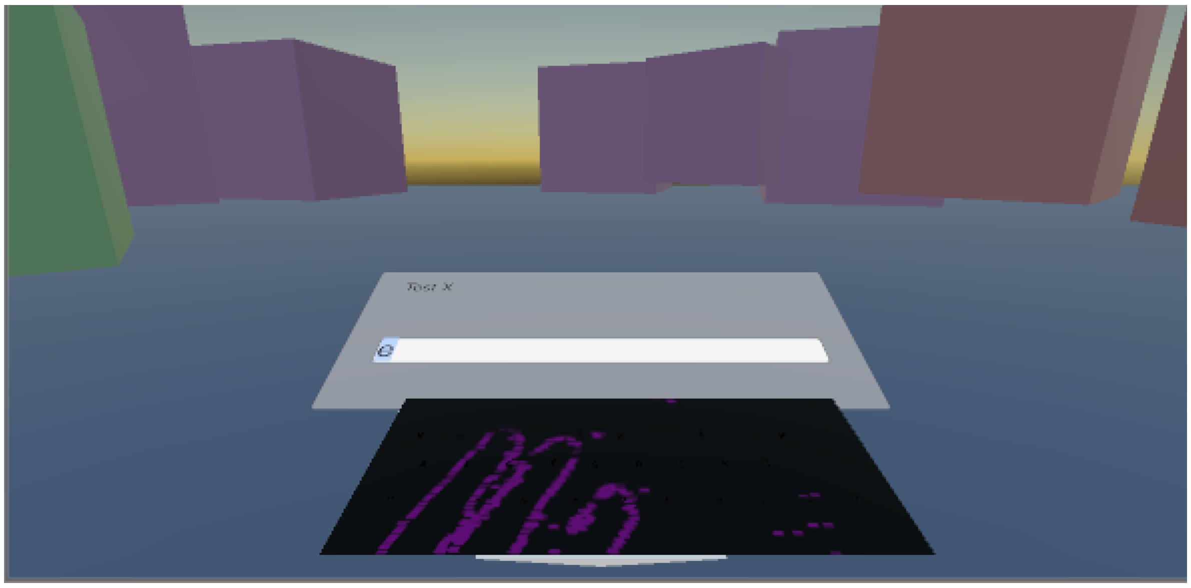
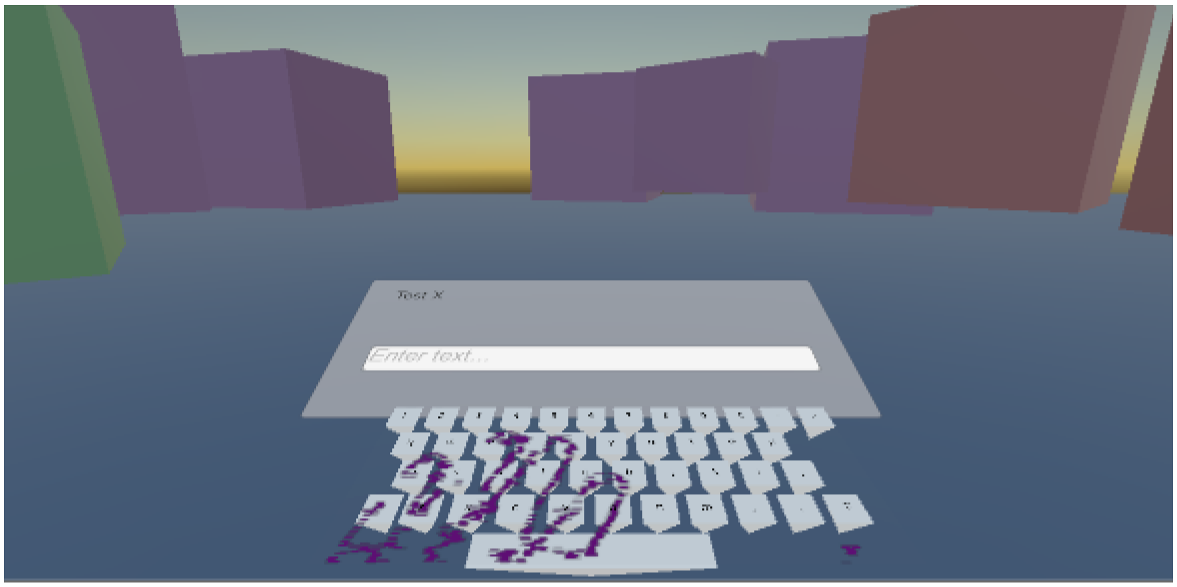


Figure 3.7: *Incorrect hand rendering is caused by data that is not passed correctly to Unity*

The cause of the distortion had to do with how the hand outline was stored. To prevent this undesirable result, I wrote new functions to store the line segments based on the OpenCV library's source code. The edge data was aggregated by as a list of line segments, not a whole polygon. Drawing line segments is harder to do in Unity than drawing polygons so I wrote more custom code to best render the hands in Unity. A $30 Unity Package is available for drawing line segments but using the 3rd party package would have restricted the ability to release the tool as a standalone package. The solution to this challenge proved easier than anticipated. The list of points created by the DLL was in the same coordinate system as the camera. This meant that showing the hand outlines in Unity was as easy as creating a new image made up of black pixels in all places except the listed points. The resulting image was applied as a texture to a plane, and updated every frame [Figure 3.8].



(a) *The texture created is made up of two colors*



(b) *The black pixels are rendered transparent, allowing for users to see the virtual keyboard*

Figure 3.8: *The texture applied to the plane provides a mapping between finger position, and the position of the lines drawn. Notice that the outline is made up of a series of short line segments*

3.3 Preperation for Testing

To answer the research questions posed by this project, an experiment was designed and executed. In addition to answering the research questions, the study conducted helped to improve upon the final design of the tool.

The experiment would perform a multivariate analysis of three typing interfaces for use with virtual reality devices. Two trials of A/B testing would be conducted in order to facilitate multiple comparisons between different typing strategies. One series of trials would look at comparing typing while wearing a head-mounted display with and without the computer vision tool. The second series of trials would study users typing abilities with a novel interface device - the drum keyboard. The typing abilities of users using the drum keyboard would be compared against those users ability to type on a keyboard while wearing an HMD. For the experiments, the control trial refers to the test conducted on a physical keyboard with no visual aid. The two experimental trials differ in which typing mechanism was measured. The “Camera” experimental trial refers to the computer vision tool that facilitates keyboard use. The “Drum” experimental trial refers to the study of the drum keyboard.

An experimental testing environment was created in Unity and made use of the SteamVR plugin. The testing environment was designed, tested, and targeted for the HTC Vive. The testing environment was a Unity scene with several cubes placed around the player in order to create a virtual space for the player to be present in while participating in the test. The minimal game-world environment was created, so that subjects would feel grounded and have reference frames when moving their heads.

In the control trial the testing scene displayed only a user interface (UI) element with a text input field and typing prompts. A virtual keyboard model was displayed in the experimental conditions but for different purposes. For the Camera trial, the virtual keyboard GameObject was placed in the scene just below the UI panel. In the drum trial, the keyboard visualization was identical to the camera trial except for the hand outlines. In place of the hand outlines, the user held the HTC Vive controllers and can see the virtual drumsticks in each hand. The virtual space can be seen in Figure 3.8b.

For the experiment, two groups of 30 typing prompts were created, one for the control trials and one for the experimental trials. Typing prompts were created in the following ranges:

- Short: Less than 20 characters —“Hooray”, “Thank you”, “talk soon”.
- Medium: 20 – 60 characters —“Kevin has been busy all day”, “When the bus came, she got on”.
- Long: 60 – 90 characters —“The two boys collected twigs outside, for over an hour, in the freezing cold!”.

These prompts can be found in Appendix C. Some prompts were complete sentences and others were fragments. Message lengths were decided based on sentence length given in English writing readability guidelines. In game communication is typically focused on short sentences [11] but in order to get an accurate measure of user typing abilities, longer sentences were also tested. Sentence prompts were collected from online typing tests and created by myself.

3.4 Methodology for Testing

Subjects were recruited via email from students enrolled in courses in the Interactive Media and Game Development program at WPI. Recruitment of individuals with an interest in game development and virtual reality was done since the target audience for the tool will have similar interests, needs, and skills. The test was approved by WPI’s IRB: HHS IRB # 00007374.

Users volunteered thirty minutes of their time to participate in the trial. Each time slot was randomly assigned either the camera category or the drum category. Only one subject was tested at a time, and I supervised each test. Before the test, subjects were given informed consent forms explaining the experience. The IRB approved forms can be found in Appendix D. Once the experiment had been explained, subjects filled out the printed pre-test survey. The pre-test questionnaire (Appendix A) allowed users to self-report their typing ability, and past experience with virtual reality.

Subjects were given the opportunity to wear the head-mounted display and to adjust the fit for comfort. While wearing the display, the virtual keyboard was brought up and subjects familiarized themselves with the setup of the tool and the positioning of the keyboard. Subjects in the drum keyboard condition were shown how the keyboard worked and given a few minutes to grow familiar with the task of typing using that mechanism. Users in the camera keyboard condition were allowed to position the keyboard on the desktop as comfortably as possible, then the tracking utility was set up by the supervisor, using a second keyboard connected to the testing computer.

The executable test was then re-launched, and users sat in front of a desk, wearing the head-mounted display. The supervisor filled in the subjects ID, age, and area of study that the subject reported on the pre-test questionnaire. This information was output by the typing program in order to match a subjects typing results with their survey responses. For users of the camera category, the supervisor aligned the camera with the keyboard that would be used by the subject. This was done so that subject could position the keyboard comfortably and so the setup of the keyboard was controlled by the supervisor, instead of depending on each subject to set up the camera for tracking.

In the virtual reality environment, subjects were also shown a paragraph of instructions that were also read aloud by the supervisor.

Your instructions are to type quickly and accurately.
You may make corrections using backspace, but remember that both speed and accuracy are important.
If you have questions complete the sentence first and push enter. Then ask the question.
You will be finished when you see the phrase ”Done”
Pay careful attention to capitilization and punctuation. Not all sentences end in a punctuation mark.
After reading these rules, let the instructor know you are ready to begin. ”

Figure 3.9: *The script users for instructing the subjects in the experiment*

Primarily these instructions served to instruct the subject to focus on both speed *and* accuracy. The instructions informed subjects that all sentences were not consistent in capitilazation or punctuation.

The computerized test collected typing data from users. The scripts used in the testing and to collect results are available on the author's github page - https://github.com/KeenanGray/. The following keyboard data was collected for each of the 30 typing prompts completed by the user:

1. Number of words in the prompt.

2. Number of letters submitted.

3. Time to complete prompt from first keystroke.

4. Words per minute.

5. Percent error (Levenshtein Distance).

6. Number of times the backspace key was pressed.

These values were outputed to a file of comma-seperated values and demarcated by subject ID. During the test, users saw a canvas with a text field, indicating the prompt, as well as an input field, where text they typed would be displayed. For longer sentences, the text would appear on two lines, however, the input field was a single line, and scrolled to accomodate more text. Using the assigned text-entry method, user typed the prompts one at a time until 30 prompts had been entered (10 prompts for each length range). Incorrect text input was not highlighted by the system. Users had to rely on comparing the input field with the given prompt to determine correctness.

When the user completed the test in the control condition, they were given the post-test questionnaire (Appendix B) to complete. After the control trial, the test was repeated for the experimental category assigned. Subjects filled out a second questionnaire based on the second interface. The post-test responses served as guidance for future changes to the tool and helped to understand each user's feelings about the technology. For example, users reported in the post-assessment whether or not they felt the HMD had interfered with their ability to type accurately.

3.5 Subject Demographics

The experiment outlined here was conducted with a total of fourteen subjects. The subjects were randomly assigned an experimental group: either the camera keyboard or the drum keyboard. Seven subjects were tested in each category. The camera group was made up of four non-touch typists and three touch-typists. In the drum category, five subjects identified as non-touch typists and two as touch-typists. Three women and eleven men were tested, in ages ranging from nineteen - twenty-nine years old. The inclusion of trained and everyday typists was meant to investigate how users of different ability levels responded to the different interfaces.

For virtual reality, it was important to measure the user experience beyond just typing performance. Interface design in virtual reality has a focus on user comfort and it often helps if the virtual interactions mimic the real world. The comfort of the typing solution and the affect of the interface on subjects was assessed through the surveys given to each subject. The main objective measures for typing were speed and accuracy. Typing speed was measured in words-per-minute (WPM) and typing accuracy was measured by calculating Levenshtein distance; the number of additions, deletions, and replacements required to transform one sentence into another.

4 Project Outcomes & Discussion

A large amount of data was collected in the course of the study. Each subject answered seven questions in the pre-test and completed two typing tests of thirty prompts each. There was also a nine question post-test for each subject. The data was corrected and aggregated in order to identify significant results. Despite flaws in the study, there were noticable differences in how subjects performed when typing with and without a virtual keyboard. Most valuably, the results of this study informed future design decisions and showed that there was potential for computer vision and physical devices like keyboards to remain relevant in the age of virtual reality.

4.1 Correcting and Combining Experimental Data

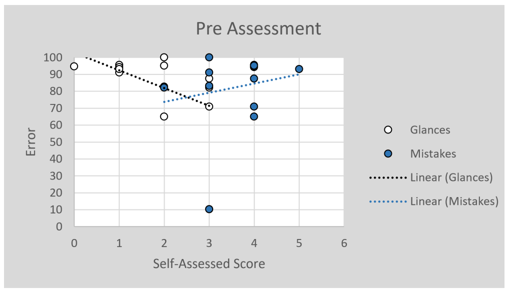
In order to compare each user independently, it was necessary to combine an individual users performance results across the thirty sentences. The comma-seperated data output by the testing program was imported to Microsoft Excel for further processing. From the collection of sentences several new data points were calculated. Each user's median WPM and median error rate was measured along with the average number of backspaces.

Correction of the data was necessary in some cases. For example, when users accidentally submitted a sentence without typing any text, the calcuations for that prompt would be inaccurate. Rows where users typed zero characters were removed from the data set. In total a dozen rows were removed across all users (1% of total) and no user had fewer than twenty-seven prompt results. Additionally, for one user information regarding backspace input was not recorded. The backspace value was used in the calculation of error so this user was excluded when analyzing error rate in the drum category.

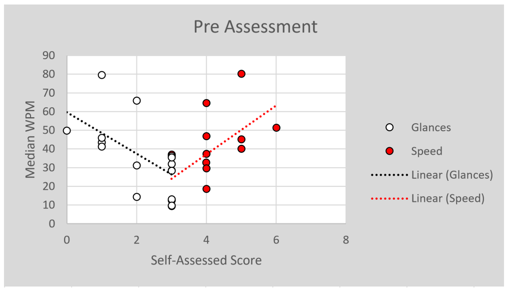
4.2 Result 1: User Prediction of Typing Ability

The first results of interest came from pre-test survey data. Users self-reported their typing ability during the pre-test. Figure 4.1 shows the results of two specific questions in the pre-questionnaire compared to the results of the control trials. In the first chart [Figure 4.1b], the WPM of a user is compared with the typing ability claimed by the user in the pre-test. A reported value of three indicates a user types "kind of slowly", a value of six indicates a user types "very fast". When the values are compared, it is clear that users self-reported ability correlated strongly with their actual typing ability in the control trial. Similarly, Figure 4.1a indicates how often users reported they made mistakes when typing relative to their correctness when completing sentences while wearing the HMD. In the questionairre, a value of one meant "makes mistakes often" and six meant "makes mistakes rarely".

Finally, users reported how often they required glances at the keyboard while typing. While no measurement was taken to see how often users actually looked at the keyboard, it is interesting to see how users assessed themselves. There is some correlation between the reported need to look at the keyboard and the outcomes of the typing test. The relationship is seen in both charts in Figure 4.1. In the questionairre, a value of three indicated a user who reported glancing at the keyboard "somewhat often", and a value of six meant a user did not feel they needed to look at the keyboard in order to type. Figure 4.1a correlates error rate and the user's reported need to see the keyboard. Figure 4.1b correlates typing speed and the user's reported need to see the keyboard. Users reporting that they required more glances at the keyboard in order to type, performed less well in terms of accuracy and speed, while wearing the head-mounted display.



(a) *A comparison between users self reported typing skill and their actual correctness.*



(b) *A comparison between users self reported typing speed and their actual speed.*

caption[Table Entry]Caption

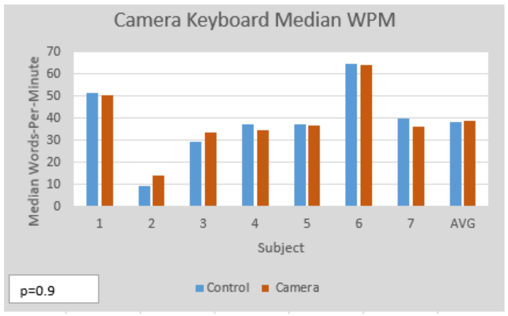
Figure 4.1: *Users' Self-Assessed performance lined up fairly nicely with resulting data.*

No typing baseline was recorded outside of virtual reality. Instead, typing was compared between guided and non-guided interfaces. Instead of a baseline, the user reported data was used to assess how much of an impact the HMD had on users ability to type. The correlation here proves that users are capable of reporting their own typing skills.

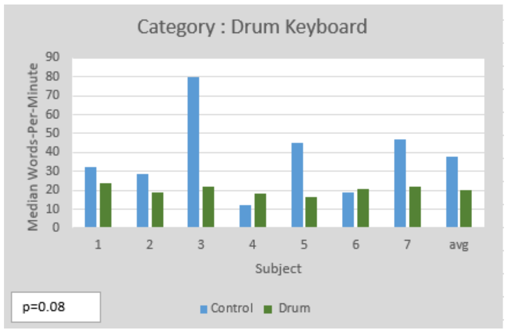
4.3 Result 2: Typing Speed Measured Across Trials

The next graphs describes user typing speed recorded in each condition. Figure 4.2 shows the differences between the WPM measured in the control and experimental trials. Figure 4.2a shows the results of each subject assigned to the camera group. Little noticable difference was observed between the control trial and the experimental trial. There is a slight overall increase in performance (a median difference of 2.5 words-per-minute) in the experimental group, but this may have resulted from the order of the trials remaining constant. By the time users began the experimental trial they may have become more adept at typing without a direct view of the keyboard. The p value of this result is quite high (0.9), implying that the results may have occured by chance.

The results seen in the second graph compare a users speed when typing in the control trial with their speed using the drum keyboard. Figure 4.2b, indicate a more substantial difference between typing speeds on a physical keyboard and on the drum keyboard. The difference here is statistically significant ($p < 0.1$) and can be explained by the fact that users had not trained previously on the drum keyboard. The drum keyboard also required greater bodily movement to use so in general was much slower, by about fifteen words-per-minute.



(a)

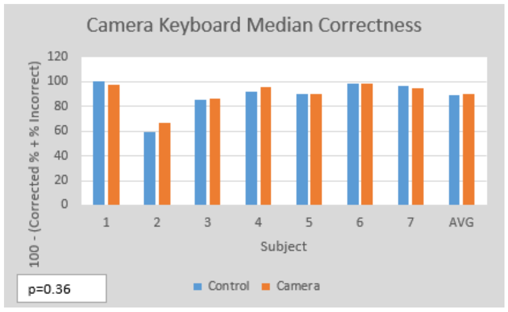


(b)

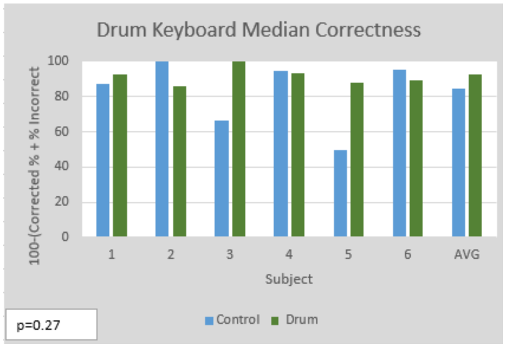
Figure 4.2: *Control and experimental results for the computer vision aid*

4.4 Result 3: Typing Correctness Measured Across Trials

The final objective measure of the experiment was correctness. Correctness was calculated by combining two of the pieces of information collected from the typing test. The percent error (CER), or Levenshtein Distance, between the sentence prompt and the sentence typed by the user was added to the percentage of the sentence corrected by the user (indicated by the number of backspace inputs / prompt length). Figure 4.3 shows the graphs for median correctness.



(a)



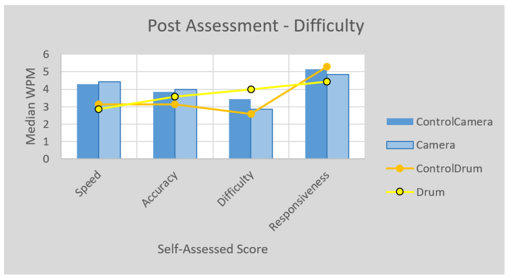
(b)

Figure 4.3: *Control and experimental results for the drum keyboard typing tool*

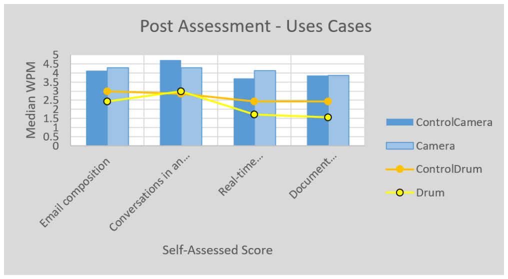
In the case of correctness, both experimental groups showed improvement. Again, because the control trial was conducted first, users may have been adapting to typing with the HMD, hence the improvement. However, subjects frequently reported that being able to see their hands and the virtual keyboard aided in finding the correct keys, as opposed to relying on memory. The reduction in error rate likely stems from the ability to see the position of the keys users intend to hit, bu there may be other factors as well. Another untested factor was the utility provided simply by having the image of a keyboard available. Some users may have memorized the position of frequently used keys, but may not have known the entire keyboard layout by memory. Having a visual guide showing the spatial relationship between keys may have helped users even without the hand-tracking tool.

4.5 Result 4: Post-Assessment Responses

The final portion of data worthy of discussion addresses the user experience of each typing mechanism. Subjective results are often important when designing human-computer interfaces. The post-survey results are presented in figures 4.4a and 4.4b. Figure 4.4a - "Post-Assessment - Difficulty" shows how users felt typing in virtual reality compared to typing with a standard monitor. For the computer vision based tool, users felt that they had typed faster and more accurately when the visual aid was provided. Additionally they found completing the task easier when their hands were tracked. For the drum category, the most notable difference was the level of difficulty of its use. Completing the task using the drum keyboard was harder for many users. Again, the difficulty of the task may have come from the requirement that more of a user's body needed to remain active to use the drum keyboard. The second graph [Figure 4.4b], "Post



(a) *Diffulty of approaches compared to control trials*



(b) *Preference for particular use cases*

Figure 4.4: *Results of post-questionaire presents information on user favorability of the interfaces*

Assessment - Use Cases" collected responses on potential applications for the typing mechanism. The most significant finding from the use case assessment shows that using a physical keyboard was highly preferable to the drum keyboard for common typing tasks. The drum keyboard seems to be too slow and uncomfortable for many users. Extended use of the drum keyboard was ound to be exhausting.

Because training on a new device is a major factor in a user's ability to type using these tools [5], the assessment of the drum keyboard was incomplete. As it is a fairly new technology, most users were unfamiliar with hand tracked controllers and some users had not used virtual reality devices previously. Nonetheless, the data collected here was valuable. Users provided feedback on the post-questionnaires about how to improve the project and many of these suggestions were implemented before the project concluded.

4.6 Result 5: Written Feedback Findings

Subjects were given space to provide written likes and dislikes after completing each trial. The feedback was evenly divided between positive and negative responses. Feedback from the control trial reflected frustrations that users had with the virtual reality headset as well as the testing environment. For example, several users complained about the text being hard to read, or the headset being uncomfortable with glasses. Subjects cited two problems with the positioning of the text prompts: the angle of the prompts, and the size. Another dislike read "The distance between my typed words and the message to be copied." referring to the gap between the sentence input field and the prompt text. The user reported that he had to move his head back and forth a lot in order to check that his input matched the text he needed to copy.

Other respones to the control trial addressed the difficulty of being unable to see the keyboard. Over half of the fourteen subjects indicated that not knowing where specific keys were affected their correctness. Users had more difficulty remembering where certain keys were, one user wrote "Disliked not being able to check where keys were" and another "Couldn't remember where keys were". This result was unanticipated since the most common challenges users would face with keyboard use was thought to be hand positioning. However, in this study, even everyday typists performed fairly well while wearing a head-mounted display. Another challenge that users faced in the control trial was submitting text with the enter key, and making corrections with the backspace key. Responses included, “Tough to adjust to mistakes” and sometimes users inadvertantly pressed “enter” when the intended another key.

For the camera trial these results were inverted. Users that struggled to find keys with no virtual keyboard liked the additional guidance provided by the tool. A particularly nice response said that the camera based keyboard “Really helps adjusting to new environment”.

The drum keyboard was positively reviewed with one major exception. Users greatly disliked that typing on the drum keyboard was cumbersome. After typing thirty prompts, users wrote that their hands, arms, or backs hurt. On a positive note, the drum keyboard was viewed as a “cool” innovation. Users enjoyed trying out new technology and practicing input with a novel tool. The haptic feedback provided by the handheld controllers was well reviewed and one user said they easily fell into a rhythm as they typed.

4.7 Result 6: Post Experiment Iteration

Several changes were made to the keyboard facilitation tool after the experiment was conducted. These changes were made based on user feedback during the test and comments left on the post-questionairre. In response to complaints that the text prompts were difficult to read, the angle of the text field and the font size were modified. The UI was repositioned and the scale of the text was increased in order to improve the legibility.

Another common complaint during testing was a lack of feedback from the on-screen keyboard. For users that touch-typed, the virtual keyboard provided little benefit. To increase the appeal of the on-screen keyboard, an animation was added to the virtual keys. When the corresponding physical key was pressed the virtual key was programmed to descend and spring back, mimicking a real keyboard. The benefits of the animation are two-fold. First, the keyboard felt more responsive since the virtual keyboard mimicked the physical behavior of the desktop keyboard. Second, the keyboard became more interactive. Instead of being a static prop, the movement hopefully engages users, making them feel less detached. Finally, the virtual keyboard model was expanded to include more keys, including ‘caps lock’,‘return’, and ‘backspace’. In hindsight it was an oversight to exclude these keys on the initial version of the keyboard used for experimental purposes. Figure 4.5 shows the tool after these changes were put into effect. T

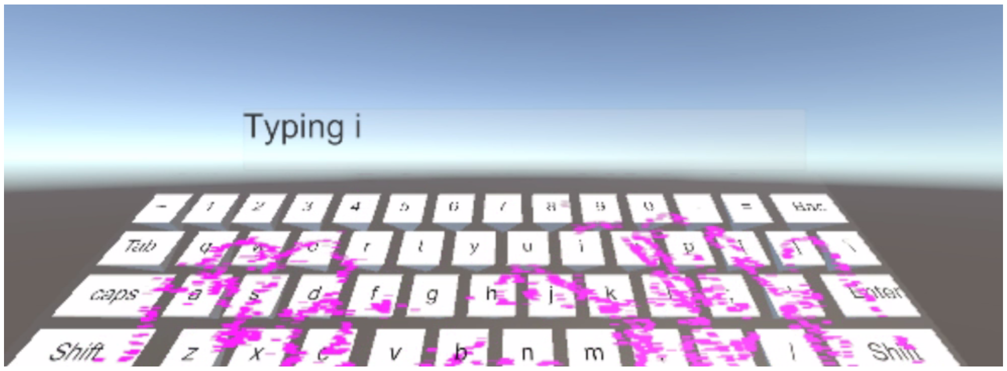


Figure 4.5: *The final tool included more keys.*

4.8 Properties of the Final Plugin

The resulting tool has minimal (close to zero) impact on the performance of the application in which it is included. The computer vision functions run in a seperate thread that is kept in sync with Unity’s game thread. The tool requires approximately 15 megabytes of space, making it sufficiently small to be included in nearly any video game project. The tool requires several dynamic libraries from OpenCV to be either installed or linked to Unity. Developers should have sufficient technical expertise to make use of the tool in their own projects fairly easily.

5 Conclusions

The final phases of this project emphasized refinement of the keyboard facilitation tool based on user feedback. Several users requested the same feature or had similar dislikes about the tested system. In some cases even a small degree of modification could have vastly improved a user's experience.

Time was a preventative factor in acquiring additional test subjects as a single test took approximately half an hour to complete. Ideally more subjects could have been tested, but the amount of testing and feedback gathered allowed for improvements to be identified and executed. The future potential of this tool or similar computer vison based interfaces appears viable. Seated experiences are a seperate category of virtual reality and offer as much potential for innovation as room-scale experiences. The VR platform is a proving ground for novel HCI designs and the keyboard can still serve an important role in the polymorphic VR space.

When looking at the directions for this project, there are avenues for future work in hardware development, software engineering, and user studies. First and foremost, finer hand tracking is likely achievable with different technology. At the moment, the software performs well, but users complained about feeling removed from the 2D projection on screen. The virtual hands felt detached and unnatural. Partial explanation of this phenomonon could be attributed to latency from the camera or the representation of the hands in the Unity program. Updating the software to be more efficient presents a solution. Additionally, a 3D model of the hands could reasonably be created dynamically from the contour data produced by the OpenCV library. Tracking markers or specialized gloves could also provide simpler means to more completely map and render a physical hand, instead of a 2D-outline, which may prove beneficial to potential users. Higher resolution stereoscopic cameras, like the Leap Motion or new sensory devices could be used to more precisely track a user's hands and improve the experience of using a tool such as this one.

Another potential area for future work could investigate the relationship between user hand placement and projection into virtual reality. One example of the way hand placement can be manipuled using virtual reality is the potential for more comfortable typing positions. An ergonomically designed keyboard could be split and located at the sides of a user's body. In VR, users will still be able to see a projection of their hand positions in their field of vision. An interface setup in this way uses virtual reality to manipulate the spaces around a user body and provide more comfortable computer interactions, without requiring a retraining of user skils.

The greatest boon of the webcamera turned out to be its portability. By applying loose constraints, the tool can be expanded to many different use cases. Text entry was the primary use case of the keyboard in this study. Future work could explore different applications for the keyboard. For example, a virtual keyboard with non-standard layout could be more easily used with the computer vision aid. Virtual Emoji keyboards, or keyboard layouts that connect more deeply with game functionality could enhance the gaming experience for many players. A character's skill bar could be displayed on the virtual keyboard, aiding in gameplay. Another example might be direct input mapping so keys on the keyboard could be used as an in-game menu system. Other desktop peripherals can easily be implemented and will function with the webcamera tracking solution. Flight joysticks for more immersive piloting games similar to *Elite Dangerous*. There's even potential for pad controllers or other computer peripherals that would be challenging to use while wearing an HMD. The scope of Unity's capabilities broadens rapidly and this tool will hopefully see use in a variety of projects.

References

[1] John Canny. “A Computational Approach to Edge Detection”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 8 (6 1986), pp. 679–697.

[2] Game Developers Conference. *The State of the Industry Report*. Mar. 2016.

[3] Joshua Corvinus. *Demonstration of a touch-type virtual keyboard in VR*. 2016. URL: https://developer-archive.leapmotion.com/gallery/vr-keyboard.

[4] Kyle Craig. *How to Implement Speech in Virtual Reality with Watson and Unity*. 2014. URL: https://www.ibm.com/innovation/milab/watson-speech-virtual-reality-unity/.

[5] David Raij Daniel Gopher. “Typing With a Two-Hand Chord Keyboard: Will the QWERTY Become Obsolete”. In: *IEEE Transactions on Systems, Man, and Cybernetics* 18 (4 1988), pp. 601–609.

[6] Anna Feit, Daryl Weir, and Antti Oulasvirta. “How We Type: Movement Strategies and Performance in Everyday Typing”. In: *CHI* (May 2016), pp. 4262–4273.

[7] A. S. Fernandes and S. K. Feiner. “Combating VR Sickness Through Subtle Dynamic Field-of-view Modification”. In: *2016 IEEE Symposium on 3D User Interfaces (3DUI)*. Mar. 2016, pp. 201–210. DOI: 10.1109/3DUI.2016.7460053.

[8] David Geerts. “Comparing Voice Chat and Text Chat in a Communication Tool for Interactive Television”. In: *Proceedings of the 4th Nordic Conference on Human-computer Interaction: Changing Roles*. NordiCHI ’06. 2006, pp. 461–464. ISBN: 1-59593-325-5. DOI: 10.1145/1182475.1182537. URL: http://doi.acm.org/10.1145/1182475.1182537.

[9] Jens Grubert et al. “Effects of Hand Representations for Typing in Virtual Reality”. In: *IEEE VR* (2018).

[10] Jens Grubert et al. “Text Entry in Immersive Head-Mounted Display-based Virtual Reality using Standard Keyboards”. In: *IEEE VR* (2018).

[11] S. C. Herring et al. “Fast Talking, Fast Shooting: Text Chat in an Online First-Person Game”. In: *2009 42nd Hawaii International Conference on System Sciences*. Jan. 2009, pp. 1–10. DOI: 10.1109/HICSS.2009.215.

[12] Thomas Hilfert and Markus König. “Low-cost virtual Reality Environment for Engineering and Construction”. In: *Visualization in Engineering* 4.1 (Jan. 2016), p. 2.

[13] Canesta Inc. *Canesta*. Screenshot; accessed April 22, 2018. 2002. URL: http://canesta0.tripod.com/paper.htm.

[14] Keith Vertanen James Walker Scott Kuhl. “Decoder-Assisted Typing using an HMD and a Physical Keyboard”. In: *CHI ’16: Extended Abstracts of the the ACM Conference on Human Factors in Computing Systems*. May 2016.

[15] Isabel Lesjak Johanna Pirker. “An Educational Physics Laboratory in Mobile Versus Room Scale Virtual Reality - A Comparitive Study”. In: *International Journal of Online Engineering* 13.8 (2017).

[16] Roger Johansson et al. “Looking at the Keyboard or the Monitor: Relationship with Text Production Processes”. In: *Reading and Writing* 23.7 (Aug. 2010), pp. 835–851. ISSN: 1573-0905. DOI: 10.1007/s11145-009-9189-3. URL: https://doi.org/10.1007/s11145-009-9189-3.

[17] Kevin S. Killourhy. “A Scientific Understanding of Keystroke Dynamics”. AAI3519874. PhD thesis. Pittsburgh, PA, USA, 2012. ISBN: 978-1-267-49950-9.

[18] Y. R. Kim and G. J. Kim. “HoVR-Type: Smartphone as a typing interface in VR using hovering”. In: *2017 IEEE International Conference on Consumer Electronics (ICCE)*. Jan. 2017, pp. 200–203. DOI: 10.1109/ICCE.2017.7889285.

[19] Cheri Li. *More than Just A Game: Communication and Community in MMORPGs*. 2006.

[20] Mark McGill et al. “A Dose of Reality: Overcoming Usability Challenges in VR Head-Mounted Displays”. In: *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*. CHI ’15. New York, NY, USA, 2015, pp. 2143–2152. DOI: 10.1145/2702123.2702382.

[21] Kyle Melnick. *Punchkeyboard is a VR Keyboard You’ve Been Waiting For*. 2017. URL: https://vrscout.com/news/punchkeyboard-vr-keyboard/.

[22] Jeff Rickel and W. Lewis Johnson. “Animated Agents for Procedural Training in Virtual Reality: Perception, Cognition, and Motor Control”. In: *Applied Artificial Intelligence* 13.4-5 (1999), pp. 343–382. DOI: 10 . 1080 / 088395199117315. URL: https://doi.org/10.1080/088395199117315.

[23] Adi Robertson. *Logitech Made a VR Keyboard Kit so You Can Yype in the Vive*. Nov. 2017.

[24] Helena Roeber, John Bacus, and Carlo Tomasi. “Typing in Thin Air The Canesta Projection Keyboard — Method of Interaction with Electronic Devices”. In: *CHI 2003: New Horizons* (2003).

[25] Sherry Ruan. *Speech Is 3x Faster than Typing for English and Mandarin Text Entry on MobileDevices*. 2016. DOI: 1608.07323.

[26] Damon L. Woodard Salil P. Banerjee. “Biometric Authentication and Identification using Keystroke Dynamics: A Survey”. In: *Journal of Pattern Recognition Research* 7 (2012), pp. 116–139.

[27] Penelope Sweetser and Peta Wyeth. “GameFlow: A Model for Evaluating Player Enjoyment in Games”. In: *Comput. Entertain.* 3.3 (July 2005), pp. 3–3. ISSN: 1544-3574. DOI: 10.1145/1077246.1077253. URL: http://doi.acm.org/10.1145/1077246.1077253.

[28] Evelyn Thomas. *Game Chat Transcription Feature Available Today in Halo Wars2 for Xbox One and Windows 10 PCs*. 2017. URL: https://blogs.msdn.microsoft.com/accessibility/2017/03/15/game-chat-transcription-feature-available-today-in-halo-wars-2-for-xbox-one-and-windows-10-pcs/.

[29] Valve. *Virtual Reality - SteamVR Featuring the HTC Vive*. [Screenshot; accessed March 14, 2018 from http://www.technobuffalo.com/wp-content/uploads/2016/04/vive-valve-advert-630x354.jpg]. 2016. URL: https://www.youtube.com/watch?v=qYfNzhLXYGc,.

[30] Greg Wadley, Marcus Carter, and Martin Gibbs. "Voice in Virtual Worlds: The Design, Use, and Influence of Voice Chat in Online Play". In: *Human–Computer Interaction* 30.3-4 (2015), pp. 336–365. DOI: 10.1080/07370024.2014.987346. URL: https://doi.org/10.1080/07370024.2014.987346.

[31] David Weibel and Bartholomäus Wissmath. "Immersion in Computer Games: The Role of Spatial Presence and Flow". In: *International Journal of Computer Games Technology* 2011, Article ID 282345, 14 pages. (2011). DOI: 10.1155/2011/282345.

[32] Max Weisel. *An Open Source Keyboard to Make Your Own*. 2017. URL: http://www.normalvr.com/blog/an-opensource-keyboard-to-make-your-own/.

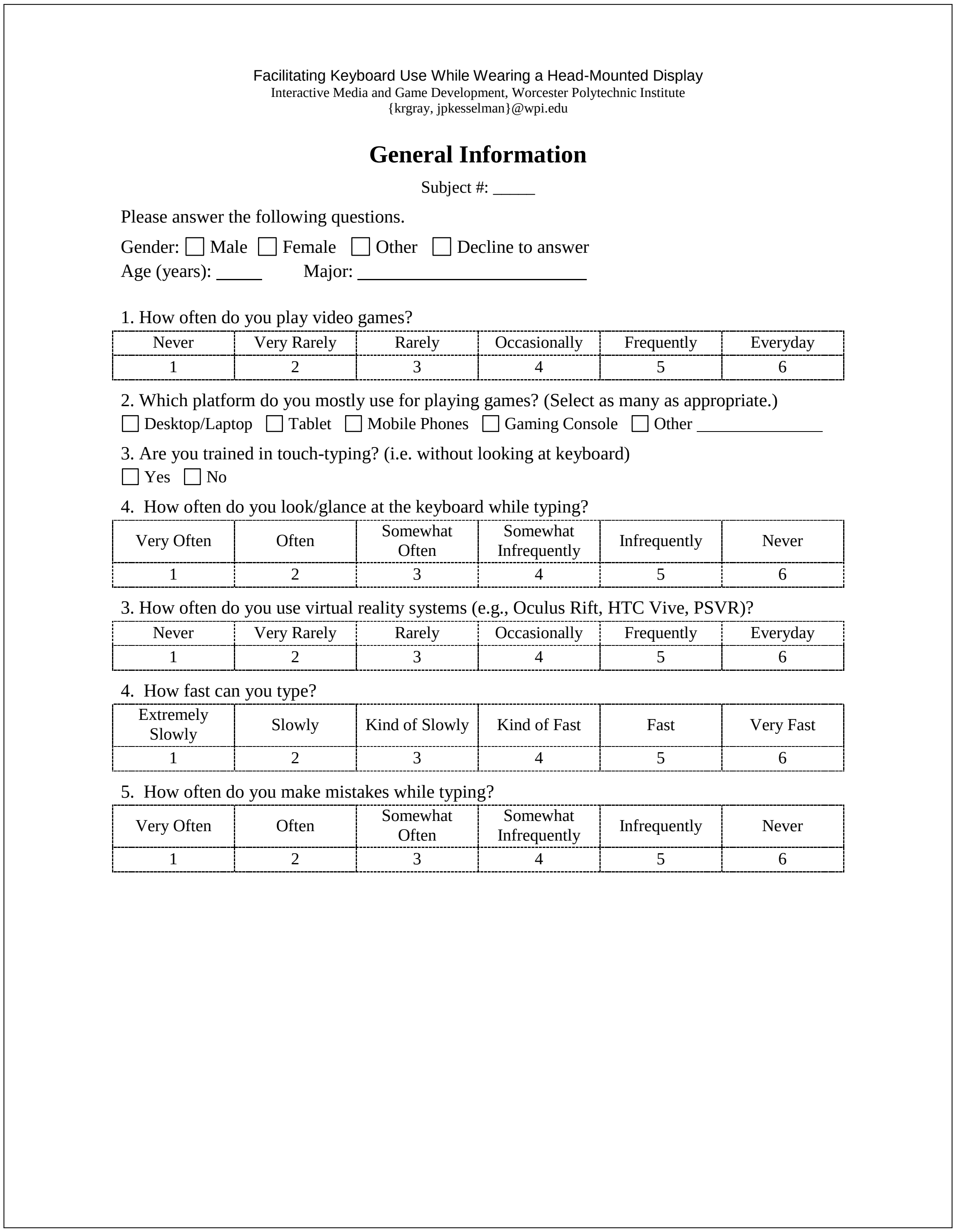
[33] Eldad Yechiam et al. "Melioration and the Transition from Touch-Typing Training to Everyday Use". In: *Human Factors* 45.4 (2003), pp. 671–684. DOI: 10.1518/hfes.45.4.671.27085. eprint: https://doi.org/10.1518/hfes.45.4.671.27085. URL: https://doi.org/10.1518/hfes.45.4.671.27085.

[34] Nick Yee. "The Demographics, Motivations, and Derived Experiences of Users of Massively Multi-user Online Graphical Environments". In: *Presence: Teleoper. Virtual Environ.* 15.3 (June 2006), pp. 309–329. ISSN: 1054-7460. DOI: 10.1162/pres.15.3.309. URL: http://dx.doi.org/10.1162/pres.15.3.309.

[35] Z. Zivkovic. "Improved Ddaptive Gaussian Mixture Model for Background Subtraction". In: *Proceedings of the 17th International Conference on Pattern Recognition, 2004. ICPR 2004.* Vol. 2. 2004, 28–31 Vol.2. DOI: 10.1109/ICPR.2004.1333992.

[36] Z. Zivkovic and Ferdinand van der Heijden. "Efficient Adaptive Density Estimation Per Image Pixel for the Task of Background Subtraction". In: *Pattern recognition letters* 27 (Jan. 2006), pp. 773–780. ISSN: 0167-8655. DOI: 10.1016/j.patrec.2005.11.005.

Appendix A Pre Questionnaire



Facilitating Keyboard Use While Wearing a Head-Mounted Display
Interactive Media and Game Development, Worcester Polytechnic Institute
{krgray, jpkesselman}@wpi.edu

General Information

Subject #: _____

Please answer the following questions.

Gender: ☐ Male ☐ Female ☐ Other ☐ Decline to answer
Age (years): _____ Major: _________________________

1. How often do you play video games?

Never	Very Rarely	Rarely	Occasionally	Frequently	Everyday
1	2	3	4	5	6

2. Which platform do you mostly use for playing games? (Select as many as appropriate.)
☐ Desktop/Laptop ☐ Tablet ☐ Mobile Phones ☐ Gaming Console ☐ Other _______________

3. Are you trained in touch-typing? (i.e. without looking at keyboard)
☐ Yes ☐ No

4. How often do you look/glance at the keyboard while typing?

Very Often	Often	Somewhat Often	Somewhat Infrequently	Infrequently	Never
1	2	3	4	5	6

3. How often do you use virtual reality systems (e.g., Oculus Rift, HTC Vive, PSVR)?

Never	Very Rarely	Rarely	Occasionally	Frequently	Everyday
1	2	3	4	5	6

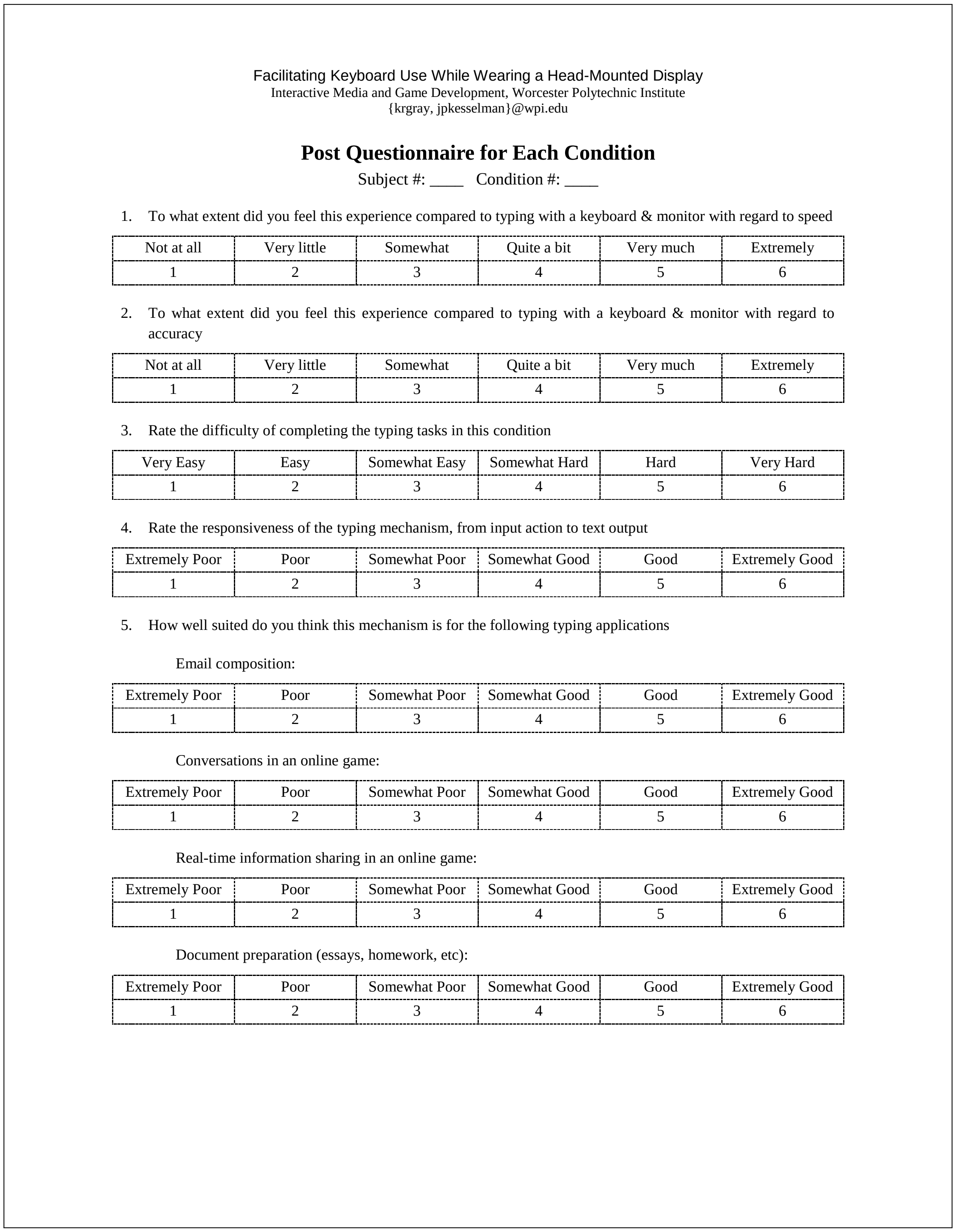
4. How fast can you type?

Extremely Slowly	Slowly	Kind of Slowly	Kind of Fast	Fast	Very Fast
1	2	3	4	5	6

5. How often do you make mistakes while typing?

Very Often	Often	Somewhat Often	Somewhat Infrequently	Infrequently	Never
1	2	3	4	5	6

Appendix B Post Questionaire



Facilitating Keyboard Use While Wearing a Head-Mounted Display
Interactive Media and Game Development, Worcester Polytechnic Institute
{krgray, jpkesselman}@wpi.edu

Post Questionnaire for Each Condition

Subject #: ____ Condition #: ____

1. To what extent did you feel this experience compared to typing with a keyboard & monitor with regard to speed

Not at all	Very little	Somewhat	Quite a bit	Very much	Extremely
1	2	3	4	5	6

2. To what extent did you feel this experience compared to typing with a keyboard & monitor with regard to accuracy

Not at all	Very little	Somewhat	Quite a bit	Very much	Extremely
1	2	3	4	5	6

3. Rate the difficulty of completing the typing tasks in this condition

Very Easy	Easy	Somewhat Easy	Somewhat Hard	Hard	Very Hard
1	2	3	4	5	6

4. Rate the responsiveness of the typing mechanism, from input action to text output

Extremely Poor	Poor	Somewhat Poor	Somewhat Good	Good	Extremely Good
1	2	3	4	5	6

5. How well suited do you think this mechanism is for the following typing applications

Email composition:

Extremely Poor	Poor	Somewhat Poor	Somewhat Good	Good	Extremely Good
1	2	3	4	5	6

Conversations in an online game:

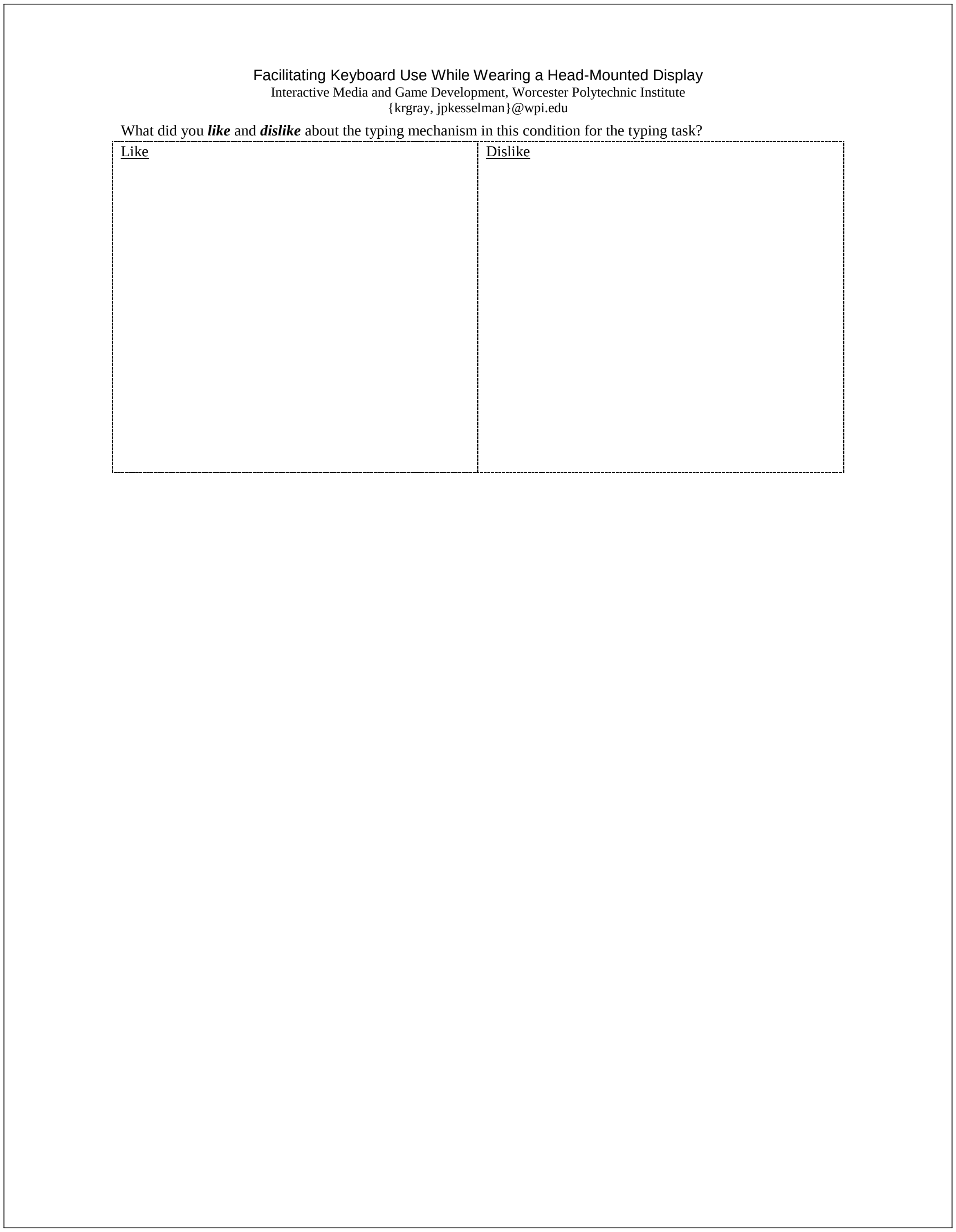
Extremely Poor	Poor	Somewhat Poor	Somewhat Good	Good	Extremely Good
1	2	3	4	5	6

Real-time information sharing in an online game:

Extremely Poor	Poor	Somewhat Poor	Somewhat Good	Good	Extremely Good
1	2	3	4	5	6

Document preparation (essays, homework, etc):

Extremely Poor	Poor	Somewhat Poor	Somewhat Good	Good	Extremely Good
1	2	3	4	5	6



Facilitating Keyboard Use While Wearing a Head-Mounted Display
Interactive Media and Game Development, Worcester Polytechnic Institute
{krgray, jpkesselman}@wpi.edu

What did you ***like*** and ***dislike*** about the typing mechanism in this condition for the typing task?

Like	Dislike

Appendix C Testing Sentences

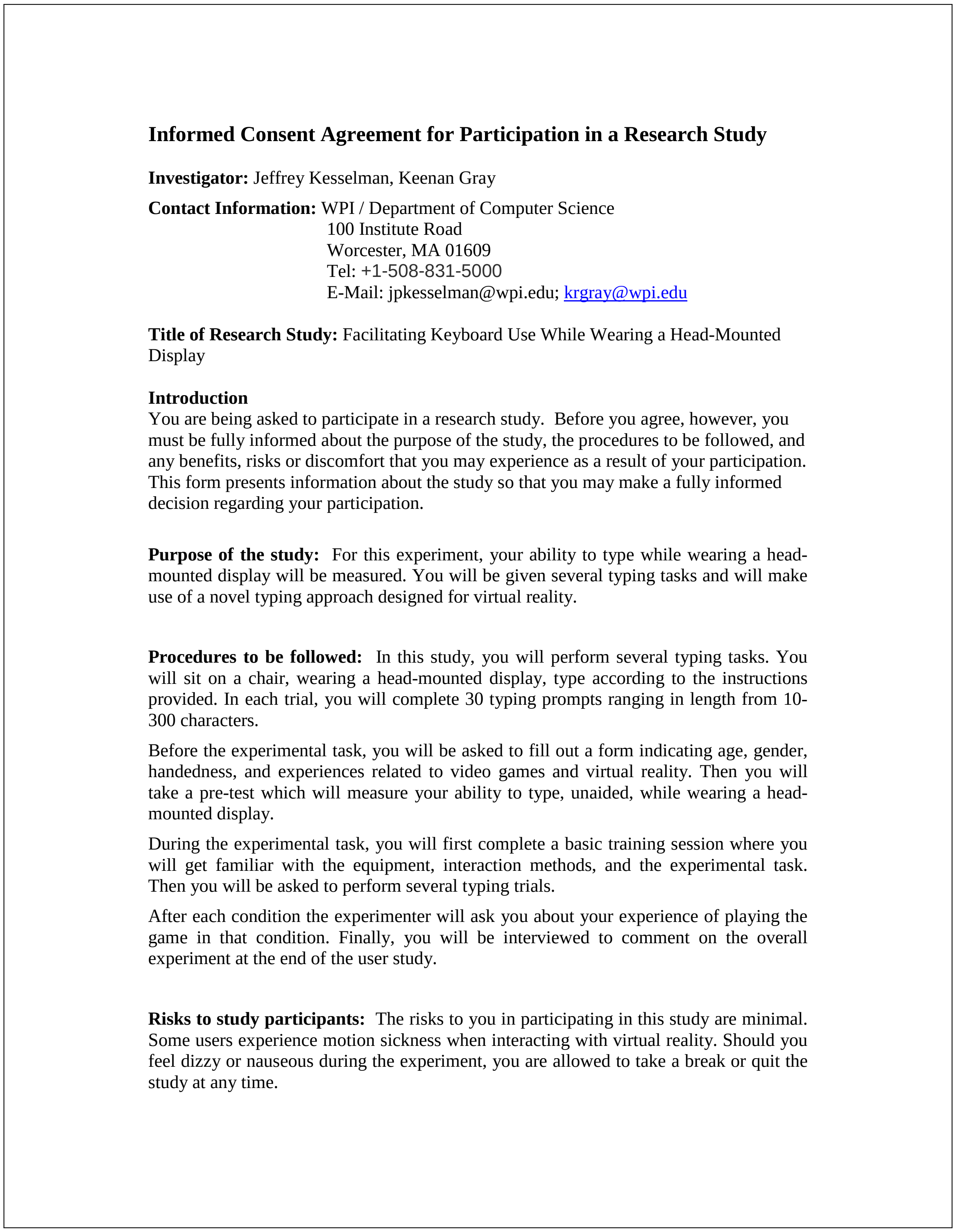
C.1 Control Test Sentences

Chat	4
Go go!	6
Thanks	6
Typing	6
Help me	7
You rock	8
Join ACM!	9
That hurt	9
I feel sick	11
Do you know her?	16
Is she in America now?	22
My son studies computers.	25
Don’t leave it up to chance	27
Which bed do you want to use	28
There has been a car accident	30
Do you think I am made of money?	32
He will finish the job by tomorrow	34
He is afraid to fly in an airplane	34
The baby was fast asleep in her mother’s arms	45
I will either write to you or phone you next week	50
In December, I went sledding every day because I could.	55
When I go swimming, I have to keep my eyes closed underwater.	61
He was evidently so busy that he even forgot to be polite to the cashier	68
You need to have a basic understanding of how things work in biology	69
All the grandfather clocks in that store were set at exactly 3 o’clock	72
That herd of bison seems to be moving quickly; does that seem normal to you?	76
Rex Quinfrey, a renowned scientist, created plans for an invisibility machine.	78
I’d like to visit Quebec, but I will have to start learning French for my trip there.	85
He continued testing the tension on his bike chain, wiping the grease on a paper napkin.	88
Despite living farther away from my family now, I make a greater effort to see them once a month	97

C.2 Experimental Test Sentences

Test	4
Goats	5
Hooray	6
I won!	6
Gotta go	8
I lost...	8
Thank you	9
talk soon	10
Are you happy?	14
Virtual Reality	15
She sang pretty well	21
Give me a bottle of wine	24
She refused my invitation	25
Kevin has been busy all day	28
When the bus came, she got on	30
Take this medicine after meals	31
She walked around looking for him	33
You talk as if you knew everything	34
They have English lessons three times a week	44
She told me which clothes would be good to wear	47
Their vocabulary is far more formal than the others.	52
This book is too expensive, I will pick out a different one	60
Those diamonds and rubies will make a beautiful piece of jewelry.	65
The London underground subway system is very efficient and easy to use.	71
Do you know why all those chemicals are so hazardous to the environment	72
Please take your dog, Cali, out for a walk, he really needs some exercise.	75
The two boys collected twigs outside, for over an hour, in the freezing cold!	77
I have three things to do today: wash my car, call my mother, and feed my dog.	79
Tom was charged with drunken drivnig after he was involved in a car accident in Boston	86
Deciding to use the sun to orient herself, she detoured around several dense patches of bushes.	96

Appendix D IRB Approved Consent Form



Informed Consent Agreement for Participation in a Research Study

Investigator: Jeffrey Kesselman, Keenan Gray

Contact Information: WPI / Department of Computer Science
100 Institute Road
Worcester, MA 01609
Tel: +1-508-831-5000
E-Mail: jpkesselman@wpi.edu; krgray@wpi.edu

Title of Research Study: Facilitating Keyboard Use While Wearing a Head-Mounted Display

Introduction
You are being asked to participate in a research study. Before you agree, however, you must be fully informed about the purpose of the study, the procedures to be followed, and any benefits, risks or discomfort that you may experience as a result of your participation. This form presents information about the study so that you may make a fully informed decision regarding your participation.

Purpose of the study: For this experiment, your ability to type while wearing a head-mounted display will be measured. You will be given several typing tasks and will make use of a novel typing approach designed for virtual reality.

Procedures to be followed: In this study, you will perform several typing tasks. You will sit on a chair, wearing a head-mounted display, type according to the instructions provided. In each trial, you will complete 30 typing prompts ranging in length from 10-300 characters.

Before the experimental task, you will be asked to fill out a form indicating age, gender, handedness, and experiences related to video games and virtual reality. Then you will take a pre-test which will measure your ability to type, unaided, while wearing a head-mounted display.

During the experimental task, you will first complete a basic training session where you will get familiar with the equipment, interaction methods, and the experimental task. Then you will be asked to perform several typing trials.

After each condition the experimenter will ask you about your experience of playing the game in that condition. Finally, you will be interviewed to comment on the overall experiment at the end of the user study.

Risks to study participants: The risks to you in participating in this study are minimal. Some users experience motion sickness when interacting with virtual reality. Should you feel dizzy or nauseous during the experiment, you are allowed to take a break or quit the study at any time.

Benefits to research participants and others: None

Record keeping and confidentiality:
Records of your participation in this study will be held confidential so far as permitted by law. However, the study investigators, the sponsor or it's designee and, under certain circumstances, the Worcester Polytechnic Institute Institutional Review Board (WPI IRB) will be able to inspect and have access to confidential data that identify you by name. Any publication or presentation of the data will not identify you.

Compensation or treatment in the event of injury: Risk of illness or injury as a result of this study is minimal. Be aware that in any event of injury or illness, you do not give up any of your legal rights by signing this statement.

For more information about this research or about the rights of research participants, or in case of research-related injury, contact: Contact the investigators Jeff Kesselman (jpkesselman@wpi.edu) or Keenan Gray (krgray@wpi.edu). Additional information can be found by contacting the IRB Chair (Professor Kent Rissmiller, Tel. 508-831-5019, Email: kjr@wpi.edu) and the University Compliance Officer (Jon Bartelson, Tel. 508-831-5725, Email: jonb@wpi.edu).

Your participation in this research is voluntary. Your refusal to participate will not result in any penalty to you or any loss of benefits to which you may otherwise be entitled. You may decide to stop participating in the research at any time without penalty or loss of other benefits. The project investigators retain the right to cancel or postpone the experimental procedures at any time they see fit.

By signing below, you acknowledge that you have been informed about and consent to be a participant in the study described above. Make sure that your questions are answered to your satisfaction before signing. You are entitled to retain a copy of this consent agreement.

___________________________ Date: ___________________
Study Participant Signature

____
Study Participant Name (Please print)

APPROVED
WPI IRB 1
2/19/18 to 2/18/19

____________________________________ Date: ___________________
Signature of Person who explained this study