Virtual Reality: An Overview, and How to do Typing in VR

Christopher Lewis*, Frederick C. Harris, Jr.*
University of Nevada, Reno, NV, 89557, USA

Abstract

Virtual Reality (VR) has existed for many years; however, it has only recently gained wide spread popularity and commercial use. This change comes from the innovations in head mounted displays (HMDs) and from the work of many software engineers making quality user experiences (UX).

Virtual Reality (VR) has many advantages compared to traditional computer interfaces; however, there are a sizable number of deficits that VR needs to solve to be more widely adopted. Arguably, the largest of these deficits is typing within VR. In the first part of this work, we present a brief history, current research areas, and areas for improvement in virtual reality. In the second part of this work, we aim to shed some insight into successful typing methods for VR through the use of a user study and a comparison of input methods. It was found that a combination of dictation and a 3D input method led to better results than solely dictation. It was also found that testing input methods with multiple types of input culminate in more varied and detailed results.

Key Words: VR; HMD; history

1 Introduction

As virtual reality (VR) continues to explode in popularity in corporate, education, entertainment, and research fields it is increasingly important to determine how VR can be used effectively. It is well known that VR can increase a user's sense of immersion and presence in a virtual environment (VE). The benefits of VR comes somewhat inherently from the medium itself, but also from a good user experience that finds its roots in core human-computer interaction principles. This paper explains portions of why VR is important, and what research has been done on its capabilities.

VR's explosion of growth has left a distinct impression on households in the U.S. Reports [2] show that 23 percent of households have used or owned a VR headset. This figure is heavily dependent on generation, with Silent Gen, Boomers, and Gen-X having 4, 6, and 18 percent having used or owned VR headsets respectively. Gen-Y and Gen-Z have 38 and 45 percent used or owned VR headsets respectively. Monthly active users for VR in 2019 was, reportedly, at 43.1 million people while in 2020 this number shot up to 52.1 million people. It is estimated that by 2030, 23 million jobs will use augmented reality and VR with healthcare, education, and blue-collar training being the most largely impacted.

The rest of this paper is structured as follows: In Section 2 we present an overview of virtual reality. This includes a brief history of Virtual Reality in Section 2.1, some of the current research areas in Section 2.2, and some areas for improvement in Section 2.2. In Section 3 we discuss typing in VR. This starts out with Section 3.2 describing a general background for typing in VR with various input methods and user studies; Section 3.3 details the implementation of the typing methods, requirements for the application, and the organization of the experiment/user study; Section 3.5 details the results obtained from the user study including WPM, EPM, SUS scores, and user responses from the post-survey; Section 3.6 discusses the results found in Section 3.5; Section 3.7 finalizes what information this work has found and gives the opinion that the way we currently measure typing speed needs to be expanded on; and Section 3.8 details expansions to this work that would further research, allow researchers to obtain better data, or would generally allow this work to expand.

2 An Overview of VR

2.1 A Brief History

While the current state of VR is dominated by commercially available head-mounted displays (HMDs) and various peripherals, an incredible amount of effort, time, resources, and research had to be invested first. VR hasn't always predominantly used HMD's, but it is where VR started. In 1960, a cinematographer named Morton Helig would receive a US patent for an invention that could show images, emit sounds, and emit air currents that could vary in velocity, temperature, and odor. This was the first known HMD. This HMD was patented under the name of: "Stereoscopic-television apparatus for individual use" and is US patent number 2,955,156. Helig produced another notable advancement in VR called the Sensorama. The "Sensorama Simulator" was patented in 1962 and displayed 3D stereo video, stereo sound, aromas, wind, and had a seat that vibrated. These inventions mark the emergence of VR. Helig also wrote in his patent about the importance of

*Department of Computer Science and Engineering. Email: christopher_le1@nevada.unr.edu, Fred.Harris@cse.unr.edu

this technology not only in a cinematographic sense, but also in: reducing hazardous situations/training for workers and the military; teaching devices to help education institutions; and multiplayer/social situations.

The next step in virtual reality quickly appeared after the Sensorama. This step came from Ivan Sutherland in the form of the "Sword of Damocles" seen in Figure 1. This invention was the first known HMD that could rotate the user's virtual field of view in tandem with how the user is physically moving their head. Dr. Sutherland's work and accomplishments are vast and could take quite a few pages of this paper. Dr. Sutherland is credited as a pioneer of computer graphics. He received the Turing award for his PhD thesis, "Sketchpad", which was the first of its kind to use a complete graphical user interface (GUI). It also influenced, if not created, modern graphical user interfaces (GUIs) and object oriented programming. Dr. Sutherland went on to create many other notable technologies and influencing many other notable students. Dr. Sutherland created "Sword of Damocles" in 1968 with a few of his students at Harvard University. The most notable students are: Bob Sproull, the former director of Oracle Labs and current adjunct professor at the University of Massachusetts Amherst; and Danny Cohen who adopted the terminology "endianness" for computing and has been inducted into the Internet Hall of Fame. The Sword of Damocles itself actually only refers to the mechanical tracking system and not the head-mounted display itself. The Sword of Damocles is also considered the first augmented reality system as the system was somewhat translucent.

From 1970 to 1990 most VR was developed for medical simulations, flight simulations, and military training purposes. A few notable inventions did occur during this time, however. In 1979, Eric Howlett created the Large Expanse, Extra Perspective (LEEP) optical system. LEEP had a wide field of view and was added to NASA's Ames Research Center in 1985. Next, Jaron Lanier founded VPL Research in 1985 where VR peripherals were being created. The most notable VR peripherals from VPL Research were the "DataGlove" and "DataSuit". The "DataGlove", was one of the first examples of a wired glove, which acts as an input device for human-computer interaction. These gloves generally mirror what the user is doing with their hands in virtual environments, though there has been some use for wired gloves to have a robot mimic what a human wearing the gloves is doing. This "DataGlove" was then licensed to companies to make entertainment related technology, most notably the "Power Glove", which was used by Nintendo in their Nintendo Entertainment System. It didn't sell well and users notoriously had a hard time with its controls and imprecision. The "DataSuit", utilizes the "DataGloves" as well as a full body suit that is filled with sensors that can measure the movement of arms, legs, and the torso.

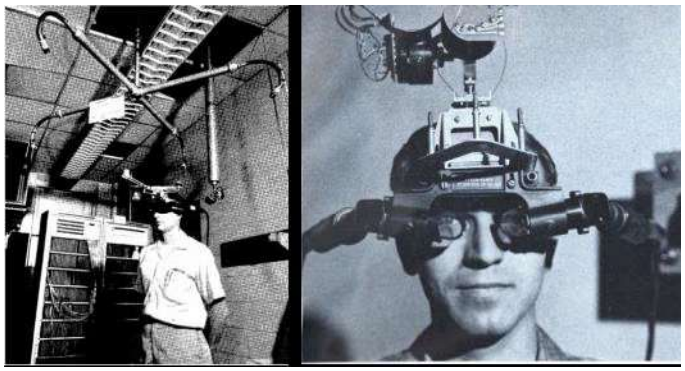

Figure 1: Ivan Sutherland's "Sword of Damocles" [28]

The 1990's saw some of the biggest changes to VR since it's inception and the seminal systems by Dr. Sutherland. One of the largest issues with VR before this point was its cost. None of these headsets were available for commercial use and they were all generally geared towards large institutions like military, medicine, or academia. In 1991 Sega announced Sega VR, which never made it to release. That same year, Virtuality launched the first mass-produced multiplayer VR systems. These systems were created for the use in VR arcades and had a cost of $73,000 per system. While not quite commercially available to the end user, this shows a distinct increase in the use of VR for entertainment and non-industry use. Again in 1991, the next large step in VR was taking place in academia.

The Cave Automatic Virtual Environment (CAVE) was a PhD thesis created by Carolina Cruz-Neira [7]. Daniel J. Sandin, a professor emeritus at the University of Illinois at Chicago, and Thomas A. DeFanti, a professor at the University of Illinois at Chicago, are also credited with creating the CAVE. The CAVE can come in quite a few forms, but the most complete CAVE is a six-sided fabric lined room using one projector and one mirror to light each side of the room, from the outside, with features from a simulated virtual environment. One example of a four-sided CAVE can be found in Figure 2.

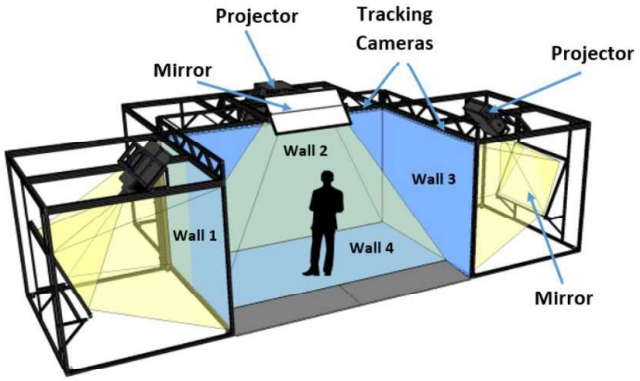

Figure 2: Annotated diagram of a four-sided CAVE system with mirrors from NASA's GRUVE Lab [19]

While this figure doesn't depict a complete six-sided CAVE, the same principles apply for any number of sides on a CAVE. This figure also shows the tracking cameras which allow the user inside of the CAVE to move around and interface with the

virtual environment in various ways. Originally this technology used electromagnetic sensors to track movements, but has since moved to infrared cameras to eliminate interference common to electromagnetic sensors. This technology has been widely adopted and you can find CAVE systems, despite their high price and long setup times, at many universities and research facilities. This technology has also evolved, as of 2012, to produce the CAVE2 [8], which is based on LCD panels rather than full projection. The CAVE2 was produced by the Electronic Visualization Laboratory (EVL) at the University of Illinois, Chicago. The two most significant aspects of the CAVE2 system are that it's cost is considerably lower than the CAVE system, and it allows for a more spherical shape to the environment, which allows for much greater realism. The CAVE2 also boasts ten times the 3D resolution of the original CAVE.

Many other inventions happened after the CAVE to produce 3D graphics, but no real progress in VR devices happened until 2010, aside from large scale VR hardware in the use of theaters and amusement rides. In 2010, the prototype for the Oculus Rift was released. This headset boasted a 90-degree field of view, which wasn't previously seen before in HMD based VR. The Oculus Rift would then be shown off at E3 and Facebook, now Meta, ended up buying it for three billion USD. Meta and Oculus later got sued by Zenimax, over the Oculus on grounds of dissemination of company secrets, who won after Meta settled out of court. The next important step in HMD innovation was done by Valve who discovered, and freely shared, low-persistence displays which make smear-free HMDs for VR possible. This technology would then be adopted by all HMD manufacturers going forward. In 2014 Sony announces Playstation VR (code name Project Morpheus). In 2015 the HTC Vive would be announced and it would use tracking technology which utilized "Lighthouses" or "base stations" that use infrared light for position tracking of the VR headset and controllers. In 2015, Google announced Google Cardboard which would bring VR to a brand new audience by utilizing smartphones for VR. Going forward, almost every large tech company either had a VR HMD released or a VR/AR group at the company.

In the current era, each of these HMDs are starting to carve their own niches in the VR space. Despite these niches, most HMDs can be categorized based off of their tracking method and connection type. Tracking methods are either outside-in or inside-out. Outside-in tracking is where the HMD, and other peripherals, are being tracked by outside sensors. Some HMDs that offer this tracking are the Oculus Rift, Valve Index, HTC Vive Pro, HTC Vive Pro Eye, HTC Vive Pro 2, and the PS VR system. Inside-out tracking is where the HMD is tracked via integrated sensors that detect changes in the position of objects in the environment. This type of tracking can be done either with or without markers placed in the room. Some HMDs that offer this tracking are the HTC Vive Cosmos, Microsoft HoloLens, and Meta Quest 2. The HTC Vive Cosmos Elite is a particularly interesting VR headset due to the fact that it allows for both inside-out and outside-in tracking due it's replaceable face plates. Connection types for HMDs mean that the HMD is either connected to the PC to be able to work, or it has a standalone system that allows the headset to work without a PC attached. Most headsets are not standalone, some that are standalone are the: Meta Quest, Pico Neo 3 Link, HTC Vive Focus 3, and HTC Vive Flow. Currently, the product line with the most flexibility and diversity is done by HTC Vive, especially now that Oculus/Meta no longer produce any HMD besides the Meta Quest 2.

2.2 Current Research

Most current research is in the software and tools for VR since creating unique and usable headsets is expensive and is already being done by large manufacturers. There is quite a bit of research dedicated to reducing or better understanding virtual reality sickness (VR sickness). There are also quite sizable projects designing applications to train or educate individuals/groups.

2.2.1 VR Sickness and Health Risks. VR sickness, which is also called cybersickness, closely resembles motion sickness in terms of symptoms. This sickness comes with a few known symptoms: Nausea, balance disorder, and vomiting. VR sickness is generally understood to be caused by a disconnect between what our senses are telling us and what is actually happening. This is called sensory conflict theory and it has been used to understand motion sickness for many years, though there are many theories relating to this topic. An example of what sensory conflict theory is presenting: If a person is in a car and looks out a window, they may get motion sickness due to the fact that their eyes see a fast moving landscape, but they personally are not moving. This would explain why some people's motion sickness gets better when they look at objects in the environment that are further away, and are thus not moving as fast due to parallax. There is also a notable conflict in the literature between deciding if gender has a role to play in VR sickness with more recent research dictating that there is no significant difference [25, 34], while later research explains that women are more susceptible to VR sickness [12, 21]. It is not clear why this conflict in the literature exists, but due to the almost ten years of time difference between the studies it could be quite a number of things.

Though not currently being researched very heavily, it has been shown that VR can be problematic for a user's sense of presence and can even induce dissociation [1]. This work shows the symptoms of depersonalization and derealization had a significant increase (4.9% - 14.5%) in their thirty participants when exposed to a virtual environment in VR. These symptoms exist to some extent in every individual, but in both the cases of participants with high and low amounts of these symptoms/feelings already, there was a significant increase in these symptoms/feelings. Due to these findings: the more realistic a virtual environment, the more careful developers need to be in showing a user unsettling and graphic things as

they could severely impact their users world view and sense of self outside of the real, objective, reality. Quite a bit more research should be put into this topic, in particular, if the dissociation issue should be listed as a part of VR sickness, meaning that they effect the same people in the same way. It would also be interesting to see if a number of factors change the amount of disassociation in the participants: the amount of realism, resolution of the environment, interactions available, multiplayer, ambient sound or background music, HMD differences, and/or locomotion techniques. It also isn't clear how long these symptoms last.

Lastly, there is another important health risk that users and developers of VR must be aware of. HMDs can be quite heavy at around 600g or 1.3lbs, so using them for extended duration and at a fast movement rate can be slightly dangerous. If the user's back, neck, or spine are sore do not continue to use the HMD. This disclaimer is brought about due to a very recent report of a German VR gamer breaking their C7 neck vertebra while playing VR [4]. This is the first ever case of a VR induced stress fracture, despite the sixty years of HMDs before this point. More cases are likely to pop up due to VR's growing proliferation, however.

2.2.2 Locomotion. Due to the fact that VR is a fairly new technology, a lot of work is being put into making it more usable and user friendly. One issue for VR is in typing, which is covered extensively in [14]. Another issue is in locomotion. Physical locomotion, which is the process of physically moving and having the VR system track and display the movement, by default isn't really that practical. Outside-in tracking causes this type of locomotion to be very limited due to the need to stay in range of a sensor. Inside-out tracking makes this locomotion a bit more plausible, but still relatively useless by itself as you're still tethered to a PC. The major downside to this locomotion technique not being usable is in the fact that physical bipedal walking comes the most naturally to humans and thus it makes VR sickness less likely. This is why redirected walking [23] was developed in 2001. Redirected walking rotates the environment around the user slowly and, almost, imperceptibly. The principle relies on the fact that humans will naturally auto-correct their movement in order to navigate through an environment. As they naturally auto-correct their physical rotation in order to direct themselves to an objective in their virtual environment, they create a curved physical path, thus increasing their available play space without even realizing it, as they only think they've walked in a straight path in the virtual environment. Many other publications have claimed they've improved on this concept using specific algorithms and machine learning [3, 13, 29], but the core concept remains the same.

Another physical locomotion technique is walking in place. This technique allows walking by having the user bring their legs up in a walking-like motion, but not actually move to anywhere physically. This technique is seldom used in favor of some of the other techniques listed below. There is an iteration of this technique that is used regularly, which is held-in-place walking or gait-negation. This technique utilizes a third party peripheral to hold the user in place as they walk or run. Two of these peripherals are the "Virtuix Omni-directional VR Treadmill" and the "Virtuix Omni One VR Treadmill". Both utilize a very slippery surface, to reduce friction of feet moving, and trackers attached to the user's shoes to detect movement. The way these two treadmills differ is in their holding mechanisms. The Omni-directional VR treadmill, uses a ring around the user that the user can lean against to move towards a virtual location. The ring itself is set to a specific height for each user before they get into the VR environment. As the user leans they slip and can walk or run forward towards that virtual location. The "Omni One" [32], as seen in Figure 3, holds the user by strapping them into a full vest that is suspended at a given height. This particular model reportedly allows the user to jump and crouch as well, something that the previous model did not. This treadmill is not currently out on the market yet, but they do have demos that make this treadmill seem very interesting for research going forward.

Figure 3: The new Virtuix Omni One VR treadmill [32]

There are quite a few other locomotion techniques. Joystick walking is where the user utilizes a joystick or a trackpad of some sort to move their avatar in a given direction. This approach can have six degrees of freedom due to the nature of the controllers. This approach also has a significant risk of VR sickness due to the fact that the user's perspective is moving, but the user themselves are not physically moving. This technique doesn't require the use of trackpads or joysticks, but some continuous actuator is required as well as some sort of vector. For example, the trigger on a controller could be the actuator and the rotation of the controller could give the angle of the vector, while the magnitude is fixed programmatically.

Teleportation based movement can come in quite a few forms. The three most prevalent forms are what we'll call: direct teleportation, preset teleportation, and avatar movement-based teleportation. Direct teleportation, in this case, refers to teleporting directly to the location that the user's cursor

is. The cursor will generally be shot out from the tip of the user's controller and fall rapidly until it collides with the floor. This then creates a target on the ground that the user can then teleport to by pressing a button. This is by far the most common teleportation method and it is the default inside of the "SteamVR Home" [31] application (the default VR launcher on a PC). "SteamVR Home" also uses preset teleportation. This teleportation allows the user to teleport only to a specific location in the environment. These locations are either single points that allow the cursor to snap directly to them, or they are larger areas that utilize a small amount of direct teleportation. The combination of direct and preset teleportation results in areas that the user can teleport inside of, but are still defined by the developers. "SteamVR Home" utilizes all of these teleportation methods by having a preset area that the user can teleport inside of and specific points in that area that the user can teleport to in order to select and change specific aspects of the virtual environment. The last teleportation method, avatar movement-based teleportation, is quite interesting. This teleportation method is the least common, but it is a very novel approach. This approach aims to solve a social VR problem, where teleportation generally feels very off putting to the people around the user teleporting. This method animates the user's avatar and makes it walk to the location the user's cursor is focused, while keeping the user's camera fixed in their current position. When the avatar makes it to their desired location, the user can then teleport their camera to that location. This means that the user temporarily sees in third person and can watch their avatar walk to a location. This locomotion technique can be seen in the application, "VR Chat" [33], which is a social VR platform. It is important to note that none of the teleportation methods can move with the six degrees of freedom that the joystick allows, but they also have significantly less risk of VR sickness associated with them.

2.2.3 Education and Training. Education in VR has always been a key field where VR shines. In recent years, training in diversity, equity, and inclusion (DEI) have been large topics of discussion. Virtual reality has been used to further this type of training to make users more engaged and present rather than simply for compliance. Technology in this specific area has been focusing on social VR, 360° video, and speech recognition, to name a few. One DEI VR application [24] attempted this type of training and met with resounding success from participants. The users in this study responded to the training's questionnaire and the researches released the following data points: 90.8% felt moderately to completely engaged; 60.5% reported feeling somewhat present or very present during the VR experience; 94.7% of respondents agreed or strongly agreed that VR was an effective tool for enhancing empathy; 85.5% agreed that the VR experience enhanced their own empathy toward racial minorities; 18.4% reported discomfort in VR; and 67.1% stated that they believed the VR experience would change their approach to communication. These survey results are overwhelmingly positive, and they validate what corporations have started to do already, which is to train their employees in DEI using VR.

Training in VR not only has the benefit of generally being more immersive and causing trainees to feel more present in their training, it also has the benefit of allowing accurate and realistic simulations and training for dangerous situations virtually. From fire simulations based in a user's home town to training for mining and construction related difficulties, virtual reality training can help save many lives and prepare individuals more accurately than many other techniques. This is also true for training in VR for medicine. One paper, depicting the results of a randomized, double-blinded study on VR training for the operating room in gallbladder surgery [26], found that VR training was 29% faster than a traditional approach and that non-VR-trained residents were nine times more likely to transiently fail to make progress. Non-VR-trained residents were also five times more likely to injure the gallbladder or burn non-target tissue. Mean errors were also six times less likely to occur for the VR-trained group.

Indeed, the effectiveness of VR training and education can be very worthwhile for quite a few fields. However, this doesn't mean that VR is correct for every situation and field. To give more understanding as to when to use VR for training, a paper titled, "Reasons to Use Virtual Reality in Education and Training Courses and a Model to Determine When to Use Virtual Reality" [20] shows insight into this problem. This paper advocates for the use or consideration of VR when: Simulations could be used; teaching or training using the real thing is dangerous, impossible, inconvenient, or difficult; a model of an environment will teach or train as well as the real thing; interacting with a model is at least as motivating as with the real thing; Travel, cost, and/or logistics of gathering the class are too high compared to using VR; shared experiences and environments are important; the creation of the environment or model is part of the learning objective; information visualization is needed; a situation needs to be made to feel real; improving perception on specific objects; developing participatory environments and activities that can't exist in the real world; teaching tasks that involve dexterity or movement; a want to make learning more interesting and fun; accessibility for disabled people; or where mistakes in the real world would be devastating or demoralizing. This paper also describes the reasons not to use virtual reality for training/education: no substitution is possible; interactions with real objects is necessary; using a VE could be physically or emotionally damaging; using a VE can result in the confusion between reality and the VE; or VR is too expensive given the learning outcome.

3 Typing in VR

3.1 Introduction

Typing is critical in any modern computer interface. Typing is also a daily occurrence for most humans on the planet today. We type in both work and leisure environments. For some people, it is their primary form of communication. For something as

critical as this, it is quite shocking that modern VR applications, generally, have little to no typing involved. When typing is involved, it is generally a slow process, especially when compared to a modern keyboard. Typing in VR often involves individually selecting letters on a virtual keyboard interface. This is a much slower process than simply typing using modern keyboards.

This work aims to speed up the typing process in VR using modern approaches and provide a methodology for reviewing input methods inside of VR. The largest addition in this project, compared to modern keyboards, is in using dictation as a primary form of input. Two input methods are created in this text. The first is using an edited form of dictation. This edited form allows the user to input any letter or, generally, any character found on a modern keyboard, as well as write whole words and sentences at a time. The second input method is a combination of an edited "drum-like keyboard" [6] and dictation. This edited form of the "drum-like keyboard" displays special characters first and allows the user to swap to an alphabetized keyboard. The reason for starting the "drum-like keyboard" with special characters selected is due to the expected use case being that the keyboard would be used to enhance dictation, not to type full sentences. The keyboard should be used to spell a single word that is hard to pronounce or hard for dictation to process. The authors created this distinction to, hopefully, facilitate understanding of when and why users might want to use the alphabetized keyboard opposed to spelling out words or acronyms using dictation.

The user study associated with this work tested for: words per minute (WPM); characters per minute (CPM); and errors per minute (EPM). The user study was broken up into four main typing challenges. These typing challenges are: URLs, Email addresses, sentences, and paragraphs. The reason these were chosen is due to the fact that they encompass almost everything the modern user of a computer might type regularly. The user study implemented a pre-test survey which collected demographic and experience data, and a post-test survey which collected general feedback and ideas for improvement. Lastly, the user study issued the System Usability Scale (SUS) for each input method, to test what the users thought about the usability of the input methods.

The rest of this section is structured as follows: Section 3.2 describes a general background for typing in VR with various input methods and user studies; Section 3.3 details the implementation of the typing methods, requirements for the application, and the organization of the experiment/user study; Section 3.5 details the results obtained from the user study including WPM, EPM, SUS scores, and user responses from the post-survey; Section 3.6 discusses the results found in Section 3.5; Section 3.7 finalizes what information this work has found and gives the opinion that the way we currently measure typing speed needs to be expanded on; and Section 3.8 details expansions to this work that would further research, allow researchers to obtain better data, or would generally allow this work to expand.

3.2 Background and Review of Literature

3.2.1 Input Methods. The classic example of VR typing methods is the "point and select" or "raycasting" method. This is a method of selecting keys on a virtual keyboard with a VR controller. The VR controller in this example sends out a raycast to the keyboard and displays itself to the user. The user would then move the controller so that the raycast hovers over a specific key and then they would press the select button to have that key input into the system. One application that uses this method is Google Earth VR [9].

Another example of VR typing methods is the "punch typing" method [35]. This is a method of selecting keys on a 3D keyboard with a VR controller. This method accomplishes typing using a collision based system. The user takes their controller, or another object that can facilitate collisions, and virtually hit the keys. The collision between the keys and the object/controller causes the key to be pressed. Keys in this method can be arrayed into a myriad of positions and can often be manually moved by the users.

The "split keyboard" or "two-thumb touchpad" [27] method is an input method for typing in VR with both controllers at the same time. Most other input methods could use both controllers at the same time, but would generally be quite uncomfortable for the user or they couldn't, generally, be used effectively. This input method aims to make use of both controllers by allowing each controller to operate distinct cursors on the virtual keyboard. This method is somewhat comparable to the "point and select" method in that they both use virtual keyboards and an, almost, cursor. However, this method is much more user friendly and much more precise. The "point and select" method is susceptible to shakes, jitters, and tremors from the user. This makes every character the user inputs a bit more of a struggle than just moving over the cursor like in the "two-thumb touchpad" method. The "point and select" method also doesn't work as well with both controllers being used to select keys as both hands would need to be extended, causing more shakes and tremors.

The "drum-like keyboard" [6] input method is almost a combination of the "point and select" method, the "punch typing" method, and the "two-thumb touchpad" method. The "drum-like keyboard" uses the VR controllers and extends a sort of drumstick (a stick with a ball on the end) out from the tips of the controllers. The 3D keyboard is displayed in front of the user where the user can then hit the keys with the drumstick, causing the key to be pressed. This allows the user to press keys with both controllers at the same time and allows for relatively quick input. The keys on the keyboard are usually at slightly different Y-coordinates based off of their row, with rows towards the user being lower than rows in the back. The keyboard itself is generally rotated towards the user as well, to allow every key to be seen at any time.

There are also a variety of input methods that do not involve the use of controllers at all. The first of these input methods is called, "dwell typing" [10]. This typing method, while not exclusive to VR, has been integrated into VR by using

the middle of the HMD as the cursor that allows the user to wait/"dwell" over any key on a 2D keyboard to select that key. These approaches tend to be designed to help those with disabilities in allowing them to type without any additional and/or expensive equipment.

"Gaze typing" allows the user to focus their gaze on any key in a 2D keyboard and dwell until the key is selected. "Gaze typing" is an evolution of "dwell typing". The most difficult part in designing these dwell based input methods is in deciding how long the user is required to dwell before making a selection. Make the dwell time too long and the WPM will go down, make the dwell time too short and the user can select incorrect keys on accident. There is quite a bit of research into what a correct dwell time is, but most research finds that it is related to the experience a user has with these systems [16]. Another interesting approach in this area is within the use of detecting the next most likely key to be pushed, and decreasing it's specific dwell time while increasing other key dwell times [18].

3.2.2 User Studies and Comparisons. VR research is steadily increasing it's use of user studies and comparison studies. As VR is becoming more of a main stay in both commercial and public use, the research surrounding its use is moving away from pure implementation details and moving towards user studies that test unique and helpful features.

A very relevant comparison study to this work is [5], which is a comparison study between four controller-based VR text-input techniques. These four input techniques are: Raycasting; drum-like keyboard; head-directed input; and split keyboard. Raycasting is the technique most commonly used in VR applications for text input and is analogous to the previously discussed "Point and Select" method. It involves raycasting from the tip of the VR controllers to hover over a 2D floating keyboard. The user would, most often, press the trigger button to input whatever key is being hovered over by the raycast. The drum-like keyboard is the same technique that this work is using, except its primary focus is on alphabetical keys in a QWERTY fashion. Head-directed input uses the rotation of the head to select specific keys and is analogous to the previously discussed "dwell typing" method. The cursor is in the middle of the user's field of view and the user would hover over the key to select it. Split keyboard input uses both controller's thumb pads to move around a cursor for each controller in a 2D keyboard that is split in half. This method is analogous to the previously discussed "two-thumb touchpad" input technique. This work found the mean WPM for each input method did not exceed twenty-one, with the drum-like keyboard being the fastest min and max WPM range. Due to this paper's findings, this work is using the drum-like keyboard as a part of the tested input methods.

One publication that details a user study that tests a unique and helpful feature is [22]. This work details a system for continuous identification and authentication of users in VR systems. This system uses a variety of body relation and movements to allow for a somewhat accurate identification system in VR based off of very simple tasks in VR. These tasks include: pointing, grabbing, walking, and typing. It trains a random forest classifier to create the identification process. They found an accuracy for each individual task and reached a highest accuracy rate of about 40%, but it would be interesting to see what the identification accuracy rate would be if they combined all stages and all tasks for an overall combined accuracy. As headsets and other VR capture devices get more accurate and allow for further granularity of motion capture, this could be a very accurate way to authenticate users.

3.3 Implementation

3.3.1 Software Engineering. The identified functional requirements for this project are found in Table 1. The identified non-functional requirements are found in Table 2.

3.3.2 Technology. This work uses Unity [30] as it's main development platform. We also used an HTC Vive Pro Eye [11] as the main HMD. Unity's built-in "DictationRecognizer" was used for basic dictation. Unity's "DictationRecognizer" is built on top of the Microsoft speech recognizer [17]. The reason the authors didn't use Microsoft Azure's Cognitive Speech Services SDK, despite Microsoft explaining it as generally better, is due to the financial trade-off for slightly better results. It was also identified that the free Unity "DictationRecognizer" would work for our purposes. The authors also used Unity's XR Interaction toolkit, version 2.0.0, for basic VR setup and OpenXR support.

3.3.3 Dictation. Dictation using Unity's "DictationRecognizer" required quite a few additions/changes to their algorithm in order for it to be used for our purposes. The goal for the dictation input method was to allow any text to be input at a user's discretion. The way this dictation recognizer works is by generating a preliminary "DictationHypothesis" and then finalizing it into a "DictationResult" when the user is done talking. The "DictationResult" had a few complications, however.

The "DictationResult" doesn't have contextualized output in most cases. The result would allow the user to say contractions (I've, Haven't, Wasn't, etc.), which shows contextualized output of special characters. For most special characters; however, the dictation algorithm would either not allow the special character to be placed or the special character would be placed, but the word would not. For example, if the user were to try to dictate "I care, period." by saying, "I care comma period period", the "DictationResult" would output as, "I care,..". This shows that the "DictationResult" changes what was verbally said into a special character. This example also shows that there would be no way to dictate the example sentence, as any instance of the word "period" would change into the special character. This works in the same way for commas. In contrast to this, the "DictationResult" would not accept other special characters in the same way. For example, if the user were to try to dictate "#Hashtag" by saying, "hashtag hashtag" or "sharp hashtag", or any other naming variation, the "DictationResult" would output exactly what the user said and not replace anything with a special character.

Table 1: The identified functional requirements

FR #	Functional Requirement Description	Priority
FR01	User can type any character found on a traditional keyboard	1
FR02	User can dictate any character found on a traditional keyboard	1
FR03	User can delete any character	1
FR04	Key presses should be clearly shown	1
FR05	Any input should be output to the same UI	1
FR06	Any input should be checked for validity	1
FR07	When a user finishes a stage the next stage should appear	1
FR08	The system will seamlessly switch between stages	1
FR09	When a new stage appears all input and sample text should be replaced	1
FR10	User can select and move to the testing scene	1
FR11	User can select and move to the main scene	1
FR12	User performance data should automatically be collected for each stage	1
FR13	User performance data should automatically output for each stage	1
FR14	When dictation is used the dictation hypothesis should be displayed	2
FR15	User can use dictation hypothesis as input text	2
FR16	Incorrect typing or dictation should be clearly shown	2

Table 2: The identified non-functional requirements

NFR #	Non-Functional Requirement Description	Priority
NFR01	User can use the controllers to collide with and type a key	1
NFR02	The system shall be well documented	1
NFR03	The system will have sample text to be used in each stage	1
NFR04	Keys should move down and then back up when pressed	1
NFR05	All input text will be managed by an intermediary writer	1
NFR06	The intermediary writer will put all input into the same UI	1
NFR07	The intermediary writer will validate input text against the sample text	1
NFR08	The intermediary writer will transmit to the system if input text is incorrect	1
NFR09	The system will support user interface interaction using the point and click method	1
NFR10	The system will use select-able buttons to move the user between scenes	1
NFR11	User performance data will be written to a file titled the start time for the program	1
NFR12	Every user interaction and dictation will be written to a full log with timestamps	2
NFR13	User can press a button to use dictation hypothesis as input text	2
NFR14	Input text and keys should turn red if an incorrect character has been entered	2

The “DictationResult” has another problem. Single characters can’t be input for things like acronyms, to spell out a particularly difficult word to pronounce, or a word that might not be a part of the dictation dictionary. For example, if the user were to try to dictate “ABC” the “DictationResult” would output, “hey be see”. Thus, not only does the “DictationResult” not take into account single letters, but it places spaces in between words by default.

To correct these issues, we created something called a “command phrase”. This “command phrase” consists of a “CommandWord” followed by a “statement”. This “CommandWord” is a word that dictation can recognize and is set at run-time by the user. The “CommandWord” for the user study was set to the word “command” for all users in order for the study to take less of the user’s time. The “CommandWord” can be said before any character to dictate an intended “statement”, which is just a character the user wishes to input. The “statement” can be any letter in the English alphabet or any special character found on a modern QWERTY layout keyboard. This implementation means that the “DictationResult” needed to be modified to remove the automatic swapping of periods and commas with their special character. For example of both changes, if the user were to try and dictate “I care, period.”, the user would need to dictate, “I care “CommandWord” comma period “CommandWord” period”. The “command phrase” in this example sentence is both ““CommandWord” comma” and ““CommandWord” period”. This example also shows that at any point in dictation, the user can input a “command phrase” along with the rest of the dictation. This same approach works with individual letters and other special characters. For cases with multiple synonyms, for example the special character “#”, which can be called a sharp,

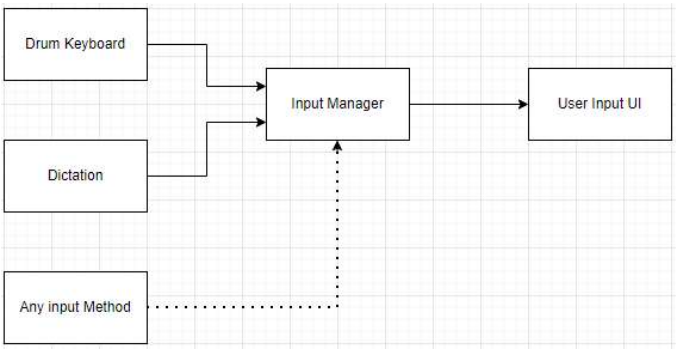

Figure 4: A block diagram detailing the basics for input method setup

pound, or hashtag, can be called by any of those synonyms and dictation will produce the special character when led by the "CommandWord". These synonyms and wording for each letter are put into a text file that is read in at the start of the program that equates the special character with the synonym. While this approach does complicate typing, it allows a VR user to type anything they wish, albeit with a bit of practice. The largest issue with this approach is if the "DictationResult" does not output a known synonym or wording for any character. For example, if the "DictationResult" output "sea" instead of "see" for the letter "c", and we didn't include that as a possible synonym, then the model would not understand and just output, " "CommandWord" sea" rather than replacing the text with "c".

To correct the automatic spacing after each word, we implemented a variant of the "command phrase". The variance comes from the "statement" portion of the "command phrase", where instead of identifying a character a user would issue a "general command". A "general command" is a word that is associated with a specific function of the input or dictation system. For example, we created a "general command" using the word "spaces". When the user issues this "general command", the dictation system no longer inserts automatic spaces between words or at the end of the "DictationResult". There are a large amount of "general command"s that could be created, but we have created only what was necessary to allow the user to use dictation for any input they'd traditionally be able to use. We have created "general command"s that allow the user to backspace, insert a space, toggle capitalization of individual letters, toggle automatic spacing, and issue a full-backspace.

3.3.4 Input Methods. As seen in Figure 4, the basic setup for input methods is quite simple. Any input method that we want to include in a comparison study or a general user study sends any input it receives into an input manager. This input manager itself has three jobs. The first is to capture any and all input received and to update the user's input text accordingly. This includes backspaces. The second job is to allow the authors to implement rather unique features that you won't find by default on a modern physical keyboard. The example for this function is the whole word deletion function, which we call a "full backspace". This is an input that we created in both input methods to allow the user to delete a full word. This is particularly useful with voice dictation as it can sometimes detect completely incorrect words. The third job of the input manager is to detect if the input is correct or not and then send out appropriate commands to other objects. This is used in the instance where the user makes a mistake when typing, or corrects a mistake when typing. If the user makes a mistake, the input text and their virtual keyboard (if they are using a keyboard) turn red. If the user goes back and corrects their mistake, the input text and their keyboard turn back to their normal color.

The edited "drum-like keyboard" (drum keyboard) displays special characters by default, but it is designed to be fully adaptable to any key set. A key set is the set containing the characters to be displayed, and used as input, for each key in order. When the application starts, the "key manager" sets every key in the drum keyboard to match a specific key set. The key manager can also be told at run-time to change the active key set. This feature is used to swap between special characters in a custom format, lowercase QWERTY-layout characters, and uppercase QWERTY-layout characters, though it could be used just as easily with keyboards for other languages besides English, as well as different keyboard layouts like Dvorak and Colemak.

3.3.5 Experiment Organization. The experimental organization falls into four different stages. One stage for each input type. Each stage is run through twice, once with each input method. These stages are picked at random to allow the data gathered to not be influenced by the familiarity of the input techniques and allows a less biased look at the data.

The application also has three distinct scenes for the experiment. The first scene that users see is the "Menu Scene". This scene gives the user the choice to select between the remaining two scenes in a user interface. The mechanism for interacting with this user interface is similar to the "point and click" or "raycasting" input method described earlier in this work, except not used on a 2D keyboard. Once selected, the application will take the user into that scene. This scene can be viewed in Figure 5.

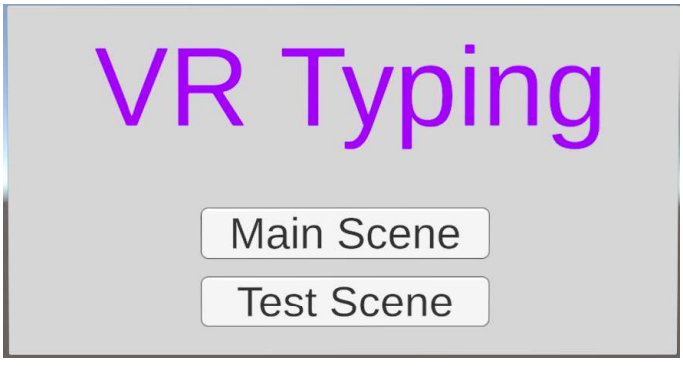

Figure 5: The menu scene

The second scene is the "Test Scene" which allows the user to use both the dictation and the drum-like keyboard input methods

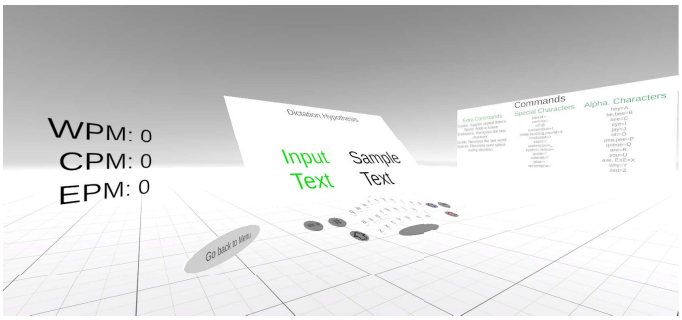

Figure 6: The testing scene

freely. This scene also displays to the user what their current words per minute (WPM), characters per minute (CPM), and errors per minute (EPM) are. Lastly, this scene displays a rotating list of sample texts that are not included in the actual experiment. This scene is to allow the user to test and play around with the input methods to get used to them before the experiment. This lowers the amount of learning that has to happen inside of the actual experiment and will allow those who aren't familiar with VR to get accustomed to it as well as the input methods. To allow the user to start the experiment, there is also a button that will take the user back to the "Menu Scene". The mechanism for pressing this button is the same as in the "drum-like keyboard". This scene can be viewed in Figure 6.

All WPM in this work is calculated using the following formula [15]:

$$WPM = \frac{|T|-1}{S} \times 60 \times \frac{1}{5}$$

where, $|T|$ is the overall resulting length of an input string, S is the number of seconds between the first and last input. $|T|$ is subtracted by 1 due to the fact that time does not start until the first input. The formula multiplies by 60 to transfer from words per second to words per minute, and it divides by 5 to transfer from characters per minute to words per minute, since a general word is considered to be 5 characters.

The third scene is the "Main Scene" this is the scene where the actual experiment takes place. This scene is very similar to the "Test Scene", except it has three main differences. The first difference is that the WPM, CPM, and EPM readings are not shown to the user and are instead averaged together per stage. Each time the user finishes a stage with an input method, this average gets output to a file and reset for the next stage/input method. This was done as a way of getting the user to try their best without any thoughts or anxiety about not performing as well as they did during the test scene. The second difference is that the return to menu button doesn't exist, so that the user can't accidentally hit it. The final difference is that the input methods rotate and the sample text is taken from a wider list of possible texts that do not include any sample text from the test scene. This scene can be viewed in Figure 7.

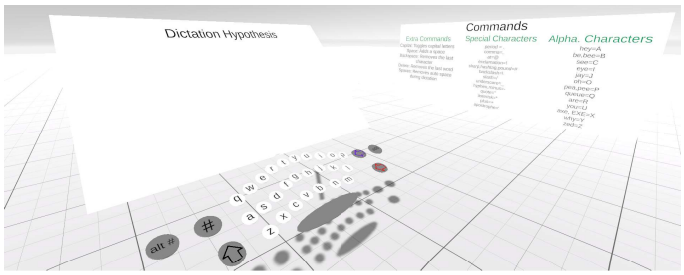

Figure 7: The main scene

3.4 Typing Text

The text that the participants had to copy involved four types of input text. This text was made out of four categories: URLs, email addresses, sentences, and paragraphs. The reasoning behind using URLs is the fact that they are very common when using a 2D typing interface, and most people are familiar with them. They also lend an interesting problem set to this process. URLs tend to need special characters, non-words, and specific abbreviations to function. This includes abbreviations like www, com, net, org, and edu. Special characters exist in URLs as well, ".", "/", and, ":". Emails also contain special characters, non-words, and abbreviations. They are also very common in the use of 2D typing interfaces. Sentences and paragraphs can contain a wider set of special characters than emails and URLs and can also test the effectiveness of typing methods for longer periods than emails or URLs. This means that both the verbal and physical input methods have to account for most/all special characters, non-words, and abbreviations. With these four input types, we believe that every use case for typing in VR and in 2D interfaces can be tested.

The sample text that is being pulled into the experiment are generated using online tools to verify consistent, or almost consistent, lengths and complexity. This text is placed into separate files to be pulled in at run-time when the user reaches any specific stage or completes the stage with one of the input methods. The input type and it's associated average character length and standard deviation is found in Table 3.

Table 3: Each input type with their associated average character length and standard deviation

Input Type	Character Length Mean (SD)
URL	10.2 (1.7)
Email	17.2 (2.2)
Sentence	37.5 (7.8)
Paragraph	305 (14.2)

3.5 Results

As seen in Table 4, both input methods implemented in this work have quite different words per minute (WPM) averages for each type of input. Paragraphs and sentences for both

Table 4: The mean WPM, with standard deviation, and WPM Range for each of the input methods in each input type

Input Method	Input Type	WPM Mean (SD)	WPM Range
Dictation	URL	7.83 (4.31)	2.47 - 17.26
	Email	5.00(3.79)	0.99 - 14.08
	Sentence	12.17(6.02)	2.414 - 19.5
	Paragraph	28.08(20.15)	10.38 - 69.11
Dictation + Drum-like keyboard	URL	7.70(4.71)	5.03 - 18.42
	Email	5.40(3.49)	2.34 - 14.45
	Sentence	18.83(13.35)	3.556 - 41.8
	Paragraph	31.69(19.28)	13.66 - 66.79

Table 5: The mean EPM, with standard deviation, and EPM Range for each of the input methods in each input type

Input Method	Input Type	EPM Mean (SD)	EPM Range
Dictation	URL	2.09 (2.02)	0.21 - 6.6
	Email	2.22(1.51)	0.6 - 5.14
	Sentence	2.41(1.78)	0.21 - 5.97
	Paragraph	3.22(2.51)	0.76 - 8.75
Dictation + Drum-like keyboard	URL	4.00(4.05)	0.22 - 13.45
	Email	3.40(3.15)	0.33 - 10.00
	Sentence	4.32(4.14)	0.55 - 10.06
	Paragraph	5.72(3.87)	1.86 - 16.38

input methods have larger WPM averages and larger standard deviations, drastically larger in the case of the dictation + drum-like keyboard input method. The email and URL input types had very similar results for both input methods. As seen in Table 5, the mean errors per minute (EPM) found for each input type are very similar to each other within each individual input method. The dictation + drum-like keyboard input method does have an average error rate and standard deviation of about twice than found with just dictation. The maximum errors per minute for this input method are also about double that of the dictation input method. The minimums are a bit different. The minimum errors per minute for the sentence and paragraph input types share the doubling found with the maximum, mean, and standard deviation; however, the email input type for the dictation + drum-like keyboard input method is about half that found in the dictation input method. The URL minimums are about the same.

The System Usability Scale (SUS) questionnaire is a measure of the usability of each system. The mean SUS score can be found in Table 6. These scores range from 0 to 100. The SUS was given to each participant twice. Once for the dictation input method and once for the dictation + drum-like keyboard input method. This table showcases that, on average, users found the combination input method of dictation + drum-like keyboard more usable. To give a better grasp on the data collected, Figure 8 and Figure 9 are included. Figure 8 showcases the SUS scores for the dictation input method. Figure 9 showcases the SUS scores for the dictation and drum-like keyboard input method.

Table 6: The mean SUS scores and SUS range

Input Methods	SUS Mean (SD)	SUS Range
Dictation	54.63 (26.54)	12.5-100
Dictation + Drum-like Keyboard	68.75 (16.81)	37.5-95

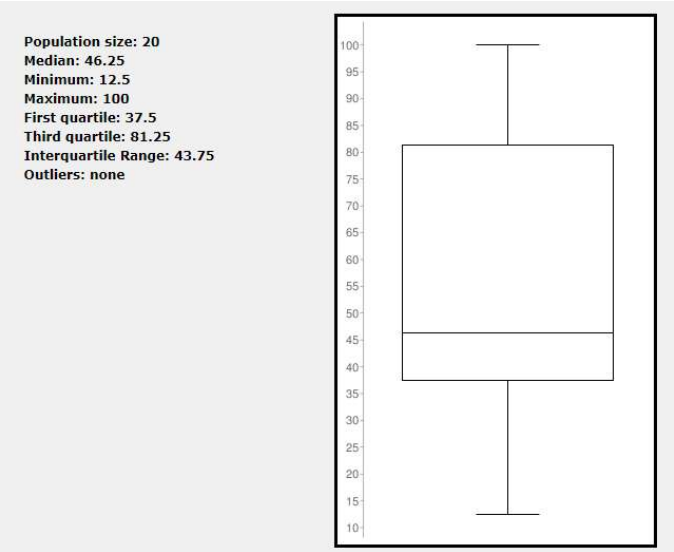

Figure 8: The box and whisker plot of SUS scores for the dictation input method

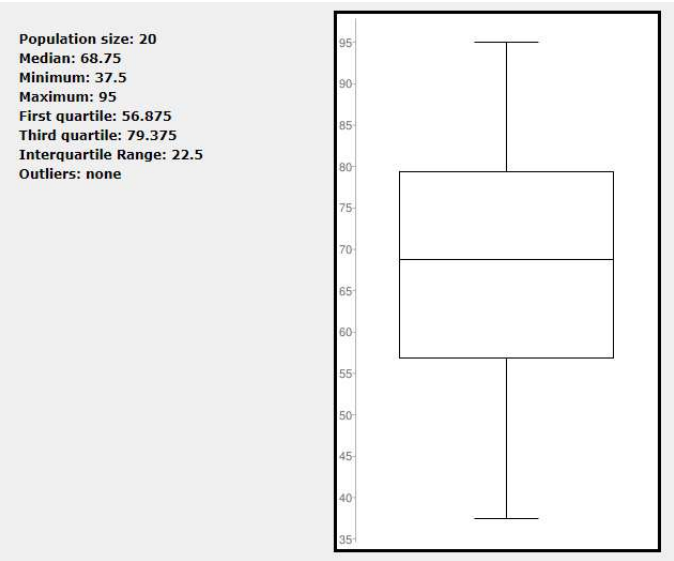

Figure 9: The box and whisker plot of SUS scores for the dictation + drum-like keyboard input method

The participants were also given a post-survey. This post-survey consisted of ten questions, which can be seen in Table 7. The first five questions were on a Likert scale from one to five. Participant responses to these five questions can be found in

Table 7: The questions asked during the post-survey

Q1	Please rate how comfortable you were during the test
Q2	Please rate how easy it was to input each text
Q3	Please rate the intuitiveness of the user interface
Q4	Please rate the usefulness of the typing methods in general
Q5	Please rate your overall experience
Q6	What potential improvements, if any, would make the application more useful or easy to use?
Q7	Based on your experience, what are some other virtual reality typing methods you would find useful?
Q8	Based on your experience, do you have a preference for any typing method?
Q9	Based on your experience, do you think any typing method was better for one input type or another (email, url, sentence, or paragraph)?
Q10	Please write any other comments or observations that you might have.

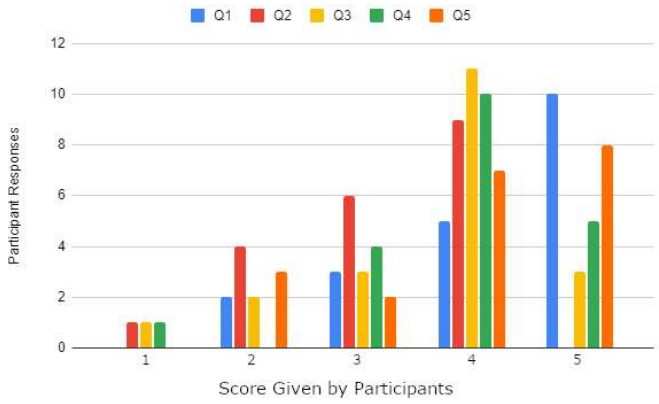

Figure 10: Participant responses for questions one through five of the post-survey

Figure 10. Users reported mostly five out of five for question one, comfortability during the test. Users reported mostly four out of five for question two, ease of text input. Users reported mostly four out of five for question three, intuitiveness of the user interface. Users reported mostly four out of five for question four, usefulness of the typing methods. Finally, users reported mostly five out of five for question five, overall experience.

Questions six to ten were free response and were then analyzed for significant statements and positive/negative sentiments for each input method and for each input method on each input type. Dictation had a total of fifteen positive statements made about it and nine negative statements. There were also eleven statements about the need to clarify or indicate some aspects of the system. The dictation method was also described as good for sentences and paragraphs, with thirteen and seventeen statements respectively. URLs and Emails received three and four positive statements respectively, for the dictation method. The drum keyboard had a total of thirteen positive statements made about it and six negative statements. There were also five statements asking for clarification or improvements in the system. The drum keyboard was described as useful for URLs and Emails, with eleven and twelve responses respectively. The drum keyboard also received two positive responses for sentences and three positive responses for paragraphs. Many of the statements in questions six to ten were responses that described the outcomes above, and many other statements gave feedback as to what needed to change and what other input methods they'd like to see.

Lastly, participants were given the Simulator Sickness Questionnaire (SSQ) to determine if typing in VR using these methods gave any excess simulator sickness. It was determined that this application does not give any excess simulator sickness, with the most common and severe symptom being eye strain and averaging to be less than mild eye strain, which is an average of less than one out of four.

3.6 Discussion

3.6.1 Demographics. Demographic data was taken directly from the pre-survey given to the participants before they entered into the VR typing application. The total number of participants was twenty. The median, and mean, age of the participants was thirty-one years of age. A total of fifteen males and five females took part in the study. The majority of participants have completed a bachelors degree. On a scale of one to five, the majority of participants were somewhat familiar with VR and responded with a three out of five or a four out of five. The exact values for this question can be seen in Figure 11. The participants were also asked, on a one to five scale, if they were familiar with motion tracking. The responses were more scattered, but most people rated a one out of five. This response shows that most participants didn't have much familiarity with typing in VR, as most VR typing methods rely on motion tracking. Lastly, the participants were asked if they were familiar with electronic entertainment and most participants rated a four, or higher, out of five, with the large majority answering five out of five.

3.6.2 Performance of Each Method. Each method performed extremely differently when comparing across input types. Many participants reported that dictation really helped with longer sentences and paragraphs, as long as they were pretty simple. The largest factor for speed in sentences and paragraphs, as well as error rate, was whether or not the dictation algorithm understood what the participant was saying. Many participants experienced unintended results due to the

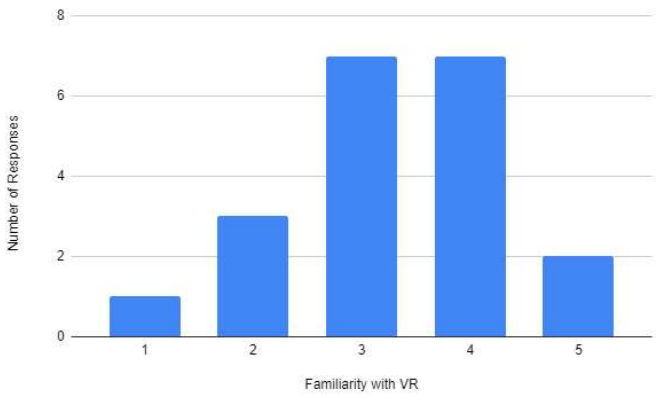

Figure 11: Participant responses for familiarity with VR

dictation results not being accurate. A large factor that hindered speed in all input types is the need to specify commands to change how the system worked. This includes commands like: "Command Spaces" and "Command Capital". A few participants reported during their post-survey that using the commands felt cumbersome, and that inputting commands fast caused the dictation result to be less accurate due to them no longer overpronouncing their dictation. Many participants also responded with the need for auto-capitalization in sentences and paragraphs. While this doesn't solve all of the problems that dictation has, it would improve accuracy, speed, and usability to a significant degree.

Participants often responded very well to the dictation and keyboard combined method. Interestingly, despite the increased usability scores and the specific mentions of enjoyment found in the post-survey, this combined input method resulted in worse errors per minute, but still faster average words per minute than the dictation method.

Another comment found in the post-survey was in the need to have more time with the systems. In particular, users found that they didn't have a large grasp of what words the dictation algorithm would insert easily and which words would require large amounts of overpronouncing. This is a common issue found with dictation algorithms as all of them, and conversational agents as a whole, can't directly tell you what they're good at. It requires a great deal of practice and trial and error to fully understand the use cases and abilities for each dictation algorithm.

Each participant was put into the test scene and were told to arbitrarily leave it when they found that they had practiced enough and had a good grasp of the input methods. This resulted in many users spending more time in the test scene than the main scene. Despite this practice; however, there was found to be no statistical correlation between time spent in the test scene and performance. For completeness, Table 8 shows the time spent in both the test and main scenes per participant, as well as average and standard deviation for time spent in both scenes.

3.7 Conclusions

The dictation and drum-like keyboard input method was overall faster and more prone to errors than the dictation input method. Many users found the drum-like keyboard fun to use in short bursts, but over a larger use period the method was found to be cumbersome. This was due to the large amounts of movement required in typing anything lengthy.

Differing input types were found to change the average speed of both methods considerably. Dictation was determined by users to be better for longer strings of real words, like sentences and paragraphs. The drum-like keyboard was determined by users to be better for precise phrases or special characters.

Due to the large discrepancy found in speeds and error rates for each input type, it is our recommendation that future work regarding text input in VR should include not only sentences or phrases, as commonly found in current research, but also; paragraphs to test long form and endurance writing, and email addresses or URLs to test short input and special character insertion. More input types can be added to fully encompass everything a user might type with a traditional keyboard. Overall, the addition of more varieties of text in researching typing methods is incredibly important as VR transitions from a novelty entertainment and scientific games platform to an interface that many people can use daily in their work.

Table 8: The time (in minutes) spent in each scene per participant

P#	Time Spent in Test Scene	Time in Main Scene
1	3.40	6.00
2	7.03	9.31
3	23.26	16.32
4	5.03	7.58
5	14.08	5.37
6	7.02	4.43
7	8.24	14.56
8	6.03	12.38
9	12.06	6.09
10	14.57	8.02
11	8.51	10.25
12	16.57	8.02
13	4.02	4.22
14	18.48	30.15
15	4.11	25.14
16	10.59	6.17
17	18.57	12.42
18	5.58	6.34
19	7.34	14.35
20	9.25	3.35
Total Time	203.74	210.47
Avg Time	10.48	9.03
Standard Deviation	5.76	3.73

3.8 Future Work

Typing in VR has always been a rather slow and non-portable process. As part of the research to fix these issues, we would like to create a generalized interface system. This system would allow the user to move around the keyboard, or any other object associated with it, in 3D space complete with six degrees of freedom. This system would also allow the user to scale and disable the interface at will. This system, therefore, would be extremely useful in applications and other research that involves both locomotion and typing intermittently.

We would also like to incorporate many input methods into this research to allow a more complete observation of input methods, particularly in the case of the SUS. With a full view of the available input methods, we believe that participants of the user study would rate each method differently on the SUS.

A longitudinal study incorporating all of the input methods would indicate what the average speed of each input method actually is. Users in this study, for instance, displayed a need to use the input methods more to fully understand the methods. Some users reported that they didn't fully understand the input methods until some of the longer input types, or towards the very end of the study. Typing in 3D is not something that most users have experienced, so letting them get used to the systems is imperative.

Compound commands are when a participant would aim to give two commands at the same time to the dictation system. This was a somewhat common mistake users would feel natural in doing. These compound commands often were caused by the want/need to have uppercase or lowercase letters. The compound commands would take the form of "'CommandWord' Capital letter", where letter is a letter they wanted to input. The proper syntax in the current system to accomplish the same thing would be, "'CommandWord' Capital" + "'CommandWord' letter". The new type of capitalization system is almost analogous to a shift key opposed to the current system which is analogous to a caps lock toggle. Implementing the compound command system would severely increase spelling speed and severely lower error rates in dictation.

With the addition of input methods, we would like to incorporate all of them into a typing game/test. This VR typing game would be complete with a virtual environment, background music, locomotion, leaderboards for each input method, multiplayer competitive scenarios, and non-player character battles.

We also feel that there can be more creative ways to allow the user to type and change input types. A few examples of extra features to incorporate into the typing for the drum-like keyboard are: using the radial touch pad on the VIVE controllers to change between keysets; colliding both controllers on a single key to change the input without completely changing the keyset; allowing the user to change, create, and save keysets dynamically; and allowing the user to use the SWYPE feature, commonly found on mobile devices, using the drum keyboard as to minimize the amount of movement per word and increase user endurance. There are also creative ways to add commands to dictation that might shorten the amount of typing necessary. A good example of this is adapting the dictation algorithm to automatically structure if statements or other programming structures, thus allowing for less key strokes and faster implementation.

Acknowledgments

This material is based in part upon work supported by the National Science Foundation under grant numbers OIA-1301726, OIA-2019609, and OIA-2148788 Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

References

[1] Frederick Aardema, Kieron O'Connor, Sophie Côté, and Annie Taillon. "Virtual Reality Induces Dissociation and Lowers Sense of Presence in Objective Reality". In "Cyberpsychology, Behavior, and Social Networking", 13:429-435, 2010. doi:10.1089/cyber.2009.0164. PMID: 20712501.

[2] ARtillery Intelligence. "VR Usage & Consumer Attitudes, Wave VI". [online]. `https://artillry.co/artillry-intelligence/vr-usage-consumer-attitudes-wave-vi/`, Last Accessed (January 10, 2023).

[3] Mahdi Azmandian, Timofey Grechkin, and Evan Suma Rosenberg. "An Evaluation of Strategies for Two-User Redirected Walking in Shared Physical Spaces". In "2017 IEEE Virtual Reality (VR)", pp. 91-98, 2017. doi:10.1109/VR.2017.7892235.

[4] D. Baur, C. Pfeifle, and C. E. Heyde. "Cervical Spine Injury after Virtual Reality Gaming: A Case Report". *Journal of Medical Case Reports*. 15, 312. May 2021. doi:10.1186/s13256-021-02880-9.

[5] Costas Boletsis and Stian Kongsvik. "Controller-based Text-input Techniques for Virtual Reality: An Empirical Comparison". *International Journal of Virtual Reality*. 19(3):2-15, August 2019. doi:10.20870/IJVR.2019.19.3.2917.

[6] Costas Boletsis and Stian Kongsvik. "Text Input in Virtual Reality: A Preliminary Evaluation of the Drum-Like VR Keyboard". *Technologies*, 7(2). 2019. doi:10.3390/technologies7020031.

[7] Carolina Cruz-Neira, Daniel J. Sandin, and Thomas A. DeFanti. "Surround-Screen Projection-Based Virtual Reality". In "Proceedings of the 20th annual conference on Computer graphics and interactive

techniques - SIGGRAPH '93", ACM Press. 1993. doi:10.1145/166117.166134.

[8] University of Illinois Chicago Electronic Visualization Laboratory. "CAVE2: Next-Generation Virtual-Reality and Visualization Hybrid Environment for Immersive Simulation and Information Analysis". https://www.evl.uic.edu/research/2016, Last Accessed (January 10, 2023).

[9] Google. "Google Earth VR". [online]. https://arvr.google.com/earth/, Last Accessed (January 10, 2023).

[10] John Paulin Hansen, Anders Sewerin Johansen, Dan Witzner Hansen, Kenji Ito, and Satoru Mashino. "Command Without a Click: Dwell Time Typing by Mouse and Gaze Selections". In "Interact", Citeseer. 3:121-128, 2003.

[11] HTC Corporation. "VIVE Pro Eye Specs". [online]. https://www.vive.com/us/product/vive-pro-eye/specs/, Last Accessed (January 10, 2023).

[12] R. S. Kennedy, M. G. Lilienthal, K. S. Berbaum, D. R. Baltzley, and M. E. McCauley. "Simulator Sickness in U.S. Navy Flight Simulators". *Aviat Space Environ Med*, 60:10–16. January 1989.

[13] Dong-Yong Lee, Yong-Hun Cho, and In-Kwon Lee. "Real-time Optimal Planning for Redirected Walking Using Deep Q-Learning". In "2019 IEEE Conference on Virtual Reality and 3D User Interfaces (VR)", pp. 63-71, 2019, doi:10.1109/VR.2019.8798121.

[14] Christopher John Lewis. *Virtual Reality Applications and Development*. Master's Thesis, University of Nevada, Reno, Reno, NV 89557. https://www.cse.unr.edu/~fredh/papers/thesis/084-lewis/thesis.pdf Last accessed: (January 10, 2023). August 2022.

[15] I Scott MacKenzie and Kumiko Tanaka-Ishii. *Text entry systems*. Morgan Kaufmann, Oxford, England. ISBN:978-0123735911. Mar. 2007.

[16] Päivi Majaranta, Ulla-Kaija Ahola, and Oleg Špakov. "Fast Gaze Typing with an Adjustable Dwell Time". In "Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, Boston, MA, USA", Association for Computing Machinery, New York, NY, USA, CHI '09. pp. 357-360, 2009. doi:10.1145/1518701.1518758.

[17] Microsoft. "Voice Input in Unity". [online]. https://docs.microsoft.com/en-us/windows/mixed-reality/develop/unity/voice-input-in-unity, Last Accessed: (January 10, 2023). May 2021.

[18] Martez E. Mott, Shane Williams, Jacob O. Wobbrock, and Meredith Ringel Morris. "Improving Dwell-Based Gaze Typing with Dynamic, Cascading Dwell Times". In "Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems, Denver, Colorado, USA", ACM, New York, NY, USA, CHI '17. pp. 2558-2570, 2017. doi=10.1145/3025453.3025517.

[19] NASA. "GVIS - GRUVE Lab - Glenn Research Center". [online]. https://www1.grc.nasa.gov/facilities/gvis/gruve-lab/, Last Accessed (08/03/2022).

[20] Veronica S Pantelidis. "Reasons to Use Virtual Reality in Education and Training Courses and a Model to Determine When to Use Virtual Reality". *Themes in Science and Technology Education*, 2(1-2):59–70. 2010.

[21] George D. Park, R. Wade Allen, Dary Fiorentino, Theodore J. Rosenthal, and Marcia L. Cook. "Simulator Sickness Scores According to Symptom Susceptibility, Age, and Gender for an Older Driver Assessment Study". In "Proceedings of the Human Factors and Ergonomics Society Annual Meeting", SAGE Publications. 50:2702-2706, October 2006. doi:10.1177/154193120605002607.

[22] Ken Pfeuffer, Matthias J. Geiger, Sarah Prange, Lukas Mecke, Daniel Buschek, and Florian Alt. "Behavioural Biometrics in VR". In "Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems", ACM. May 2019. doi:10.1145/3290605.3300340.

[23] Sharif Razzaque, Zachariah Kohn, and Mary C. Whitton. "Redirected Walking". In "Eurographics 2001 - Short Presentations", Eurographics Association. 2001. doi:10.2312/egs.20011036.

[24] Robert O. Roswell, Courtney D. Cogburn, Jack Tocco, Johanna Martinez, Catherine Bangeranye, Jeremy N. Bailenson, Michael Wright, Jennifer H. Mieres, and Lawrence Smith. "Cultivating Empathy Through Virtual Reality: Advancing Conversations About Racism, Inequity, and Climate in Medicine". *Academic Medicine*. 95:1882-1886, July 2020. doi:10.1097/acm.0000000000003615.

[25] Dimitrios Saredakis, Ancret Szpak, Brandon Birckhead, Hannah A Keage, Albert Rizzo, and Tobias Loetscher. "Factors Associated with Virtual Reality Sickness in Head-Mounted Displays: A Systematic Review and Meta-Analysis". *Frontiers in Human Neuroscience*. December, 2019. doi:10.3389/fnhum.2020.00096.

[26] Neal E. Seymour, Anthony G. Gallagher, Sanziana A. Roman, Michael K. O'Brien, Vipin K. Bansal, Dana K. Andersen, and Richard M. Satava. "Virtual Reality Training Improves Operating Room Performance". *Annals of Surgery*. 236:458-464, October 2002. doi:10.1097/00000658-200210000-00008.

[27] Jeongmin Son, Sunggeun Ahn, Sunbum Kim, and Geehyuk Lee. "Improving Two-Thumb Touchpad Typing in Virtual Reality". In "Extended Abstracts of the 2019 CHI Conference on Human Factors in Computing Systems, Glasgow, Scotland UK", Association for Computing Machinery, New York, NY, USA, CHI EA '19. pp. 1-6, 2019. doi:10.1145/3290607.3312926.

[28] Ivan E. Sutherland. "A Head-Mounted Three Dimensional Display". In "Proceedings of the December 9-11, 1968, fall joint computer conference, part I on - AFIPS '68 (Fall, part I)", ACM Press. 1968. doi:10.1145/1476589.1476686.

[29] Jerald Thomas and Evan Suma Rosenberg. "A General Reactive Algorithm for Redirected Walking Using Artificial Potential Functions". In "2019 IEEE Conference on Virtual Reality and 3D User Interfaces (VR)", pp. 56-62, 2019. doi:10.1109/VR.2019.8797983.

[30] Unity Technologies. "Unity Real-Time Development Platform: 3D, 2D VR & AR Engine". [online]. `https://unity.com/`, Last Accessed (January 10, 2023).

[31] Valve Corporation. "SteamVR Unity Plugin". [online]. `https://valvesoftware.github.io/steamvr_unity_plugin/`, Last Accessed (January 10, 2023).

[32] Virtuix. "Omni One — The Future of Gaming". [online]. `https://omni.virtuix.com/`, Last Accessed (January 10, 2023).

[33] VRChat Inc. "VRChat Hompage". [online]. `https://hello.vrchat.com/`, Last Accessed (January 10, 2023).

[34] Michael L. Wilson and Amelia J. Kinsela. "Absence of Gender Differences in Actual Induced HMD Motion Sickness vs. Pretrial Susceptibility Ratings". *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*. 61(1):1313-1316, 2017. doi:10.1177/1541931213601810.

[35] Powen Yao, Vangelis Lympouridis, Tian Zhu, Michael Zyda, and Ruoxi Jia. "Punch Typing: Alternative Method for Text Entry in Virtual Reality". In "Symposium on Spatial User Interaction, Virtual Event, Canada", Association for Computing Machinery, New York, NY, USA, SUI '20. 2020. doi:10.1145/3385959.3421722.

Christopher Lewis received his BS and MS degrees in Computer Science and Engineering from the University of Nevada, Reno in 2020 and 2022 respectively. During this time he specialized in Virtual Reality as a researcher in the High Performance Computation and Visualization Lab. Since graduating, he has went on to work as an engineer for Moth + Flame, a company making high quality VR training experiences in both hard and soft skills. He has plans to return to academia for a PhD in a few years.

Frederick C. Harris, Jr. received his BS and MS degrees in Mathematics and Educational Administration from Bob Jones University, Greenville, SC, USA in 1986 and 1988 respectively. He then went on and received his MS and Ph.D. degrees in Computer Science from Clemson University, Clemson, SC, USA in 1991 and 1994 respectively.

He is currently the Associate Dean for Research in the College of Engineering and a Foundation Professor in the Department of Computer Science and Engineering and the Director of the High Performance Computation and Visualization Lab at the University of Nevada, Reno. Since joining UNR, he has worked on research projects funded by federal agencies (NSF, NASA, DARPA, ONR, DoD) as well as industry. He is also the Nevada State EPSCoR Director and the Project Director for Nevada NSF EPSCoR. He has published more than 300 peer-reviewed journal and conference papers along with several book chapters and has edited or co-edited 14 books. He has had 14 PhD students and 81 MS Thesis students finish under his supervision. His research interests are in parallel computation, simulation, computer graphics, and virtual reality. He is also a Senior Member of the ACM, and a Senior Member of the International Society for Computers and their Applications (ISCA).