Arc-Type and Tilt-Type: Pen-based Immersive Text Input for Room-Scale VR

Bret Jackson
bjackson@macalester.edu
Macalester College
Saint Paul, Minnesota

Logan B. Caraco
Macalester College
Saint Paul, Minnesota

Zahara M. Spilka
Macalester College
Saint Paul, Minnesota

Figure 1: ***Left:*** **A user manipulates a pen-shaped stylus to enter text in a virtual reality CAVE, where traditional keyboard interaction limits mobility.** ***Center:*** **The Tilt-Type interface uses small finger motions to tilt the stylus along two axes, changing the selected character in a binned layout.** ***Right:*** **The Arc-Type interface uses a rotational gesture and interactions with a touch slider to select characters from a radial layout.**

ABSTRACT

As immersive, room-scale virtual and augmented reality become more utilized in productive workflows, the need for fast, equally-immersive text input grows. Traditional keyboard interaction in these room-scale environments is limiting because of its predominantly-seated usage and the necessity for visual indicators of the hands and keys potentially breaking immersion. Pen-based VR/AR interaction presents an alternative immersive text input modality with high throughput. In this paper, we present two novel interfaces designed for a pen-shaped stylus that do not require the positioning of the controller within a region of space, but rather detect input from rotation and on-board buttons. Initial results show that compared with Controller Pointing text entry techniques, Tilt-Type was slower but produced fewer errors and was less physically demanding. Additional studies are needed to validate the results.

CCS CONCEPTS

• **Human-centered computing** → **Text input**; **Virtual reality**; **Mixed / augmented reality**.

KEYWORDS

Virtual Reality, text entry, pen-based interaction

ACM Reference Format:
Bret Jackson, Logan B. Caraco, and Zahara M. Spilka. 2020. Arc-Type and Tilt-Type: Pen-based Immersive Text Input for Room-Scale VR. In *Symposium on Spatial User Interaction (SUI '20), October 31-November 1, 2020, Virtual Event, Canada.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3385959.3418454

SUI '20, October 31-November 1, 2020, Virtual Event, Canada

ACM ISBN 978-1-4503-7943-4/20/10.
https://doi.org/10.1145/3385959.3418454

1 INTRODUCTION

Immersive environments using virtual or augmented reality (VR or AR) are increasingly used for training, visualization, gaming, and creative pursuits [40, 43]. The technology has improved and become affordable. Indeed, a 2017 industry survey found that "... virtual reality has arrived: it works! It is mature, stable, and, most importantly, usable" [4]. However barriers to full adoption remain.

In particular, symbolic or text input remains a challenge. People are quite skilled at using keyboards in traditional desktop computing, but when wearing a head-mounted display (HMD) the lack of visual feedback makes typing more difficult [41, 56]. Recent work has focused on exploring alternative virtual representations of the keyboard and hands [25, 34] and using mixed-reality systems to overlay camera data of the user's hands with virtual content [31]. Yet these approaches require the user to be seated and do not work for room-scale VR or AR, the scenarios with the highest potential immersion and use of physical navigation to enhance presence.

As the breadth of room-scale VR applications continues to grow, so too does the need for immersive, fast, and low-effort textual interfaces that can be used while standing or moving around. Existing controller-based VR/AR text input methods (e.g. raycast selection[50]) suffer from two primary issues: (1) They are more tiring than keyboard-based techniques, and (2) They are much slower.

Unlike a keyboard where a user can rest their palms on a desk, VR/AR text input techniques often utilize spatial positioning of controllers to select individual characters, leading to increased fatigue and lower writing speed. The design of commonly available VR/AR controllers exacerbates this issue. Commonly clutched with

the whole hand in a power grip, spatial movement of the controller requires movement of the wrist or entire arm, using much larger muscle groups than simply typing keys on a keyboard.

Three-dimensional user interfaces (3DUI) using a pen-shaped stylus may address these issues. Pen-based interaction has already proven at least as efficient as a mouse, which is in turn more efficient than a standard VR controller in Fitts' law selection tasks [45]. However, attempting to use a pen-based input system while holding a tablet (e.g. Virtual Notepad [47]) produces high levels of arm strain [9]. Using the pen alone to write in midair is inefficient because of low character recognition accuracy [21].

Nevertheless, the fine-motor control and increased selection accuracy of a pen-based input device still provide benefits for alternative text input techniques, particularly if only used for simple text inputs like search queries, entering login information, or chatting. In this paper, we present two novel interfaces, *Tilt-Type* and *Arc-Type*, for text input using a pen-shaped stylus controller in a VR CAVE or room-scale VR environment. Our primary contributions are an exploration of the design space for pen-based text input and development of the two interfaces. Although the Covid-19 pandemic limited us from performing a formal in-person evaluation, we also contribute an analytical discussion of the opportunities and obstacles presented by each interface, lessons learned, and anecdotal evidence with an expert user comparing the interfaces to the commonly used raycasting-based interface.

The paper begins by describing related work on text input techniques for immersive environments. Then, we present the design and implementation of two novel input techniques. The paper closes with a discussion of a preliminary evaluation, lessons learned, limitations, and conclusion.

2 RELATED WORK

Text input has been extensively studied. For an in-depth exploration of previous interfaces and how their performance compares, see Dube and Arif's 2019 survey [15], Speicher et al.'s 2018 selection-based text entry comparison [50], or Boletsis and Kongsvig's controller-based comparison [5]. Here, we present an overview of the predominant interface types and a discussion of the most relevant pen-based text input techniques.

2.1 Speech and Physical Keyboards in VR/AR

Speech recognition is a natural mode of text entry for room-scale virtual environments, and recent systems (e.g. SWIFTER [46]) have been able to obtain speeds up to 23.6 words per minute. However, several issues can impede its use. Noisy environments, accents, and speech impediments can cause recognition errors [26] that are hard to correct [53]. Use in public environments or with multiple users can have privacy and obtrusiveness issues [26]. Most importantly, it might interfere with other cognitive processes in a way that physical interfaces like Tilt-Type and Arc-Type do not [49].

Physical QWERTY and AZERTY keyboards present an alternative. They have been and continue to be the most efficient and ubiquitous ways to input text at a console, and studies exploring their use in VR have uncovered a similar result. Most users have engaged with these layouts for the majority of their life, and therefore minimal training or adjustment is needed to regain 60% or more of a user's desktop speed and accuracy [33].

There are two primary concerns with utilization of physical keyboards in VR; use of a head-mounted display (HMD) occludes the user's view of their hands and keyboard, and keyboards are usually impractical in a room-scale VR environment [44].

Several researchers have worked to address the first concern. Walker et al. [56] developed an auto-correct decoder based on Velocitap [55] to compensate for errors. Others have studied the impact of providing virtual avatar hands [34], representations showing positions of the fingertips [25], and using AR to provide blended views of the user's hands with the virtual content [41]. The results of this work are clear: providing additional visual feedback improves performance when typing on physical keyboards in VR/AR; however, this comes at the expense of working in front of a green screen [25, 41] or augmenting the hands with tracking markers [26, 34].

Of note, the HawKEY interface [44] avoids these tracking issues by using a standard RGB-D camera and allows for greater flexibility in room-scale applications through the use of a portable keyboard worn by the user. However, users complained that the straps holding the keyboard were uncomfortable and that ergonomically it was too close to the body.

In contrast to these interfaces, our work does not purport to match a standard keyboard's performance for word processing. Instead, the Tilt-Type and Arc-Type interfaces presented in Section 3 are designed to satisfy the need to make quick and simple text interactions while physically moving, e.g. searching for a file to load, writing simple annotations about the environment, or chatting with dispersed users. In these contexts, the lack of speed may be compensated by auto-correction and -completion, as well as the brevity of the text to input. In these common, fast-use cases, the inconvenience of leaving immersion to sit at a desk is potentially greater than that of learning a novel interface.

2.2 Gesture and Controller VR/AR Text Input

The desire for fast, easy, and immersive text input is not new, and there have been many attempts at novel input interfaces. The most successful of these interfaces rely on 'hunt-and-peck' character selection using ray casting based on head pointing [59, 60] or controller pointing [50]. These techniques are able to achieve about 15 words per minute (WPM) [50], but by using Swype algorithms where the gesture path serves as input rather than selecting individual characters, experts are able to perform up to 34 WPM [11, 32].

Other approaches have used pinching between the thumb and different knuckles (e.g. BlueTap [13]) or between the thumb and fingertips (e.g. Pinch Keyboard [9] and PinchType [18]) to select characters. Pinch Keyboard is particularly worth noting for this project because it uses a rotation gesture of the user's wrist to select a set of characters followed by pinching a digit to their thumb to select one of four characters in the set. This inspired a similar two step process for our Arc-Type interface where the user first selects a letter using a rotation gesture followed by progressively refining the character choice.

In addition to raycasting and pinch interfaces, VR wand controllers have been explored for text input in VR/AR. For example, a

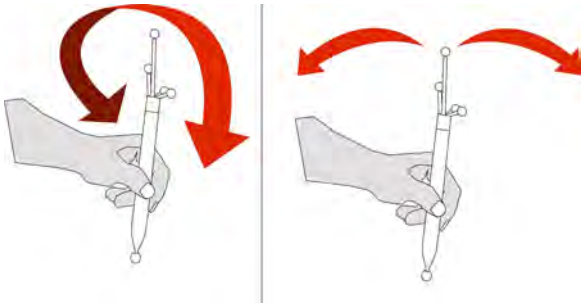

Figure 2: The Tilt-Type interaction. *Left:* Rotating the stylus left/right relative to the user's body changes the selected bin of characters (shown in Figure 3). *Right:* Pitching the stylus toward/away from the body, changes the selection within a bin.

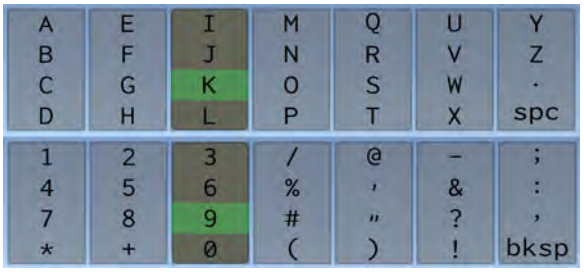

Figure 3: The standard (top) and alternative (bottom) 7x4 grid layouts for Tilt-Type. The columns are visually grouped into 'bins' of characters. The alternate layout is triggered by the secondary stylus button.

drum-like keyboard [6] allows a user to tap keys. HiPad [30] uses the circular touchpad on a common VR controller to select characters laid out in a circle. PizzaText[61] similarly uses a pair joysticks (as on a modern game console controller). One joystick selects a 'slice' of four characters, and the other designates which character to choose within the slice. HiPad and PizzaText are similar to our interfaces in that they try to exploit smaller fine-motor control of the thumb rather than large arm movements to improve control and lower fatigue. However unlike the game controller used in PizzaText, the form factor of the pen-based controller for Tilt-Type and Arc-Type can be used for more 6DoF (degrees-of-freedom) interactions besides text input. While the VR wand used in HiPad can provide similar additional 6DoF interactions, prior studies have found that pen-based controllers outperform wands for selection tasks [3, 36, 45].

2.3 Pen-Based VR/AR Text Entry

Compared with other types of input devices, there has been relatively little work on pen-based text entry techniques for VR/AR. Directly writing in the air has low character recognition accuracy, leading to many errors [21]. As a result, most pen-based interfaces use a pen and tablet approach, where the tablet records the interaction. The Virtual Notepad [47] uses a 3D-tracked tablet for simple hand-written annotation, although an optical character recognition algorithm is needed to use the output for anything other than viewing by other people. 'Connect the Dots' [20] attempts to avoid this issue by using a grid of dots that the user connects with a pen to form alphanumeric letter shapes, but a preliminary evaluation shows that this approach is still error-prone.

Bowman et al. [9] compared a pen-and-tablet-based keyboard to speech, Pinch Keyboard, and a chord keyboard. The pen-and-tablet keyboard was found to have moderately fast input with the fewest errors, and they advocate for more research on pen-based interfaces.

All of the approaches that combine pen-and-tablet input require the user to hold an extra device which can be cumbersome and lead to additional fatigue and the Gorilla arm effect [7]. However, given the increased pointing accuracy and fine motor control of a pen compared with standard wand-based VR controllers [44], we explore alternative approaches for pen-based text input that do not require a tablet or suffer from character recognition issues.

3 PEN-BASED VR/AR TEXT INPUT BEYOND WRITING

In this section, we present the design and implementation details for two 3DUIs designed to take advantage of pen or stylus-shaped input devices like the OVR Stylus [28], LogitechVR Ink [37], or Massless Pen [12]. The interfaces were designed and tested for use in a stereoscopic four-wall VR CAVE display or with a Windows Mixed Reality HMD, but they are broadly applicable to any VR or AR display configuration.

3.1 Tilt-Type

Tilt-Type is designed to leverage the fine-motor control of the fingers when gripping the stylus input device like a pen. Similar to the process of writing — where small finger movements change the orientation of a pen — Tilt-Type utilizes similar small finger movements to tilt the stylus along two axes to select characters. This approach draws parallels with TiltText [57] which uses tilting of a mobile phone to disambiguate letter choices from a traditional 12 button mobile phone and TiltWriter [10] which uses mobile device tilt to roll a ball over a virtual keybard to select letters.

The interface displays character choices in a horizontally-aligned, binned layout, shown in Figure 3. Each bin contains four characters organized alphabetically. The process of selecting a character is shown in Figure 2. The user grips the stylus like a pen, holding the barrel somewhat vertically. Rotating it slightly left or right selects a bin (which darkens). Tilting the back of the stylus away from the user chooses a character within the selected bin (highlighted in green). A button press on the stylus confirms the character choice and adds it to the text output.

The design of the interface was chosen to rely only on the angular orientation change of the stylus rather than positional changes (e.g. in raycasting approaches). This allows a user to hold the stylus comfortably at their side when performing the interaction, lowering the potential for fatigue. The visual highlighting shown on the layout for the selected bin and character allow the gesture to be performed accurately without directly viewing the stylus.

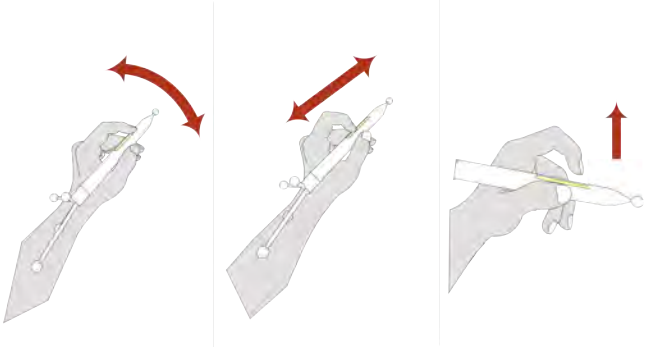

Figure 4: The Arc-Type interaction. *Left:* pivoting the stylus changes the selected bin in the radial layout (shown in Figure 5). *Center:* Sliding the index finger on a touch sensor changes the character selection within a bin. *Right:* Releasing the finger confirms the selection.

Through iterative testing, rotation and tilt angle ranges were chosen to minimize the movement needed while still providing a large enough angular change between bins and characters to make selection accurate. The bin-selecting rotation gesture ranges between [30 – -10] degrees from vertical for left-handed usage. The ranges swap to [10 – -30] degrees for right-handed usage to make it more ergonomic. The tilt gesture to select a character within a bin ranges from [5 – 30] degrees from vertical.

Many VR/AR text interfaces only support simple alphabetic characters [44]. In Tilt-Type, an alternative bin layout (shown in Figure 3) is provided to support more complex inputs that may be necessary. The alternative layout is activated by pressing a secondary button on the stylus. The bins are modified to contain numbers and punctuation. The alternative layout also supports a backspace button to remove previously output characters and correct errors. Characters in the alternative layout are chosen using the same gestures as the standard layout. Pressing the secondary stylus button a second time toggles back to the standard alphabetic layout.

3.2 Arc-Type

While Tilt-Type explores the use of fine-motor control provided by holding a pen-shaped stylus, we are also interested in exploring the design space of combining fine-motor finger movements on the pen with larger motions of the user's wrist — similar in style to how a user might use fine-motor control when writing, but larger wrist or arm motions to underline or cross out text.

Inspired by the Pinch Keyboard [9], which uses a wrist rotation gesture to pick character bins followed by finger pinches to select characters within a bin, Arc-Type uses a similar interaction. The graphical layout (shown in Figure 5) presents character bins displayed radially in an arc. Like Tilt-Type, an alternative layout containing numbers and punctuation is toggled by a secondary button on the stylus.

The user interaction is shown in Figure 4. By rotating their wrist, a user rotates the stylus around the z-axis. Angular change is mapped to bin selection within the layout. After selecting a bin, the user slides their index finger forward or backward along the touch sensor available on many common VR/AR styluses (e.g. OVR Stylus [28], LogitechVR Ink [37], or Massless Pen [12]) to select a character within the bin. Releasing the touch sensor confirms the selection.

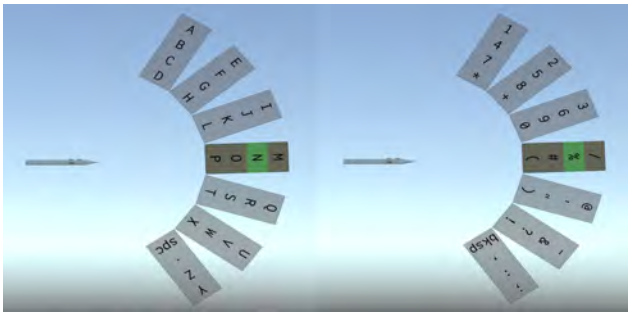

Figure 5: The standard (left) and alternative (right) layouts for Arc-Type. Seven bins arranged radially each contain four characters. A virtual stylus representation is fixed to the center of the arc and its orientation updates with angular changes of the physical stylus to aid visual feedback.

Through iterative testing, the bin-selecting rotation is constrained between [45 –145] degrees from vertical to be comfortable to the user. It is important to emphasize that the bin selection is based on angular changes of the stylus not raycasting, despite showing a virtual stylus at the center of the bin arc to give additional feedback for the currently-selected bin. This allows the visual feedback to be spatially positioned disjointly from the stylus' physical location so that the user can comfortably perform the gesture in a more relaxed arm state.

3.3 Implementation

Tilt-Type and Arc-Type were implemented in C# using the Unity3D [52] game engine. They were tested using a four-walled VR CAVE display driven by a single computer with two 3.5 GHz Intel Xeon E5-2637 processors and three NVidia Quadro P5000 graphics cards. Multiple networked instances of Unity3D are run (one per display) to provide the stereoscopic rendering using the MinVRUnity library [29].

Shown in Figure 6, an implementation of the open-source VR stylus (OVR Stylus) [28] was created to be used as a pen-like input device. A touch potentiometer slider runs down the tip end of the stylus body, with two buttons along the side. The button closer to the tip is mapped to confirming character selections in Tilt-Type and the second button, closer to the back, is mapped to toggling

Figure 6: The OVR Stylus [28] used in the implementation.

the alternative layouts. Reflective marks on the tip and back of the stylus are tracked using an OptiTrack [27] motion capture system communicating with Unity3D through the VRPN library [51]. This provides a full 6DoF (six degrees-of-freedom) controller.

Because the slider on the stylus is implemented with a potentiometer it reports touch values as an integer in the range 0 to 64 as finger pressure changes the electrical resistance along its length. Pressing close to the back of the stylus reports values near 0 and pressing the slider near the tip reports values near 64. For the Arc-Type interface, changes in the slider value are mapped to changes in character choice. When the finger is removed from the potentiometer, the electrical resistance quickly drops to zero. This behavior is used to detect the finger release action that confirms character selection in Arc-Type. To gain more precision in detecting this event and disregarding noise, a dead-zone is introduced at the end farthest from the tip of the stylus, consuming the values 0-8 as invalid. When the reported value crosses this threshold a gradient-ascent algorithm is used to detect the previous value that was reported when the finger released.

4 EVALUATION AND DISCUSSION

In this section we report on several types of results. Unfortunately, due to the Covid-19 pandemic we were unable to run an in-person study to evaluate the interfaces, and relatively-low adoption of pen-based controllers precluded us from running a distributed study. Instead we report an analytical discussion of our own use of the Tilt-Type and Arc-Type interfaces. The results include both qualitative insights, such as observations, lessons learned, and refinements made during iterative development, as well as quantitative metrics recorded during use. We start by describing an additional interface that was implemented for comparison, and finally the results.

4.1 Comparison Interface

To compare Tilt-Type and Arc-Type to state-of-the-art interfaces, we implemented a Controller Pointing interface, based on the interaction defined by Speicher et al. [50] which was found to outperform the other methods that they studied. Unlike the original Controller Pointing, our implementation was modified to only use a single 6DoF stylus controller rather than two traditional wand controllers. Shown in Figure 7, a ray is cast from the tip of the stylus highlighting the selected character in a QWERTY layout. Pressing the front button on the stylus confirms a selection.

4.2 Phrase Set

Phrase sets for evaluating text input techniques fall into two categories: (1) simple sentences containing only single-case alphabetical letters or spaces, or (2) complex sentences mixing case and including numbers or more complex symbols. Although all interfaces tested include the ability for complex sentences, we chose to evaluate using only simple sentences to facilitate comparison with other recent VR text input studies that do likewise [5, 41, 50, 56]. Based on the commonly used Mackenzie phrase set [38], 36 simple sentences were chosen at random. This smaller subset (36 of the 500 possible phrases) was used to make the study completable in a reasonable time period. The mean sentence length is 26 characters (SD = 4.17). The mean number of words per phrase is 4.97 words (SD = 0.87).

Figure 7: The Controller Pointing interface used for comparison. A ray is cast from the tip of the stylus to select characters in a QWERTY layout.

Table 1: Summary of quantitative results. Standard deviation shown in parentheses.

Interface	Blind WPM	Perfect WPM	PER	Angular Travel	PIT
ArcType	2.8 (0.20)	1.6 (0.54)	23. (8.0)	8.6E3 (5.5E3)	9.6E2 (6.6E2)
TiltType	5.3 (0.37)	4.6 (0.50)	8.6 (4.6)	1.5E3 (3.2E3)	4.0E3 (7.5E2)
Raycast	8.8 (0.88)	6.6 (1.1)	15. (6.9)	—	—

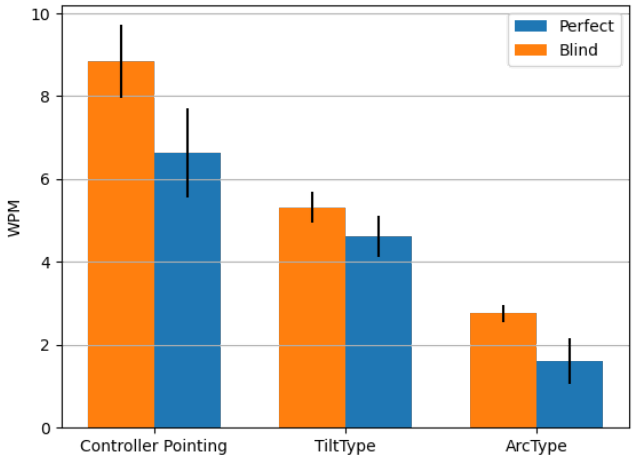

Figure 8: Entry speed (WPM) results for blind and perfect trials. Bars indicate mean value and the error bars are one standard deviation.

4.3 Metrics and Results

The interfaces were tested by one of the authors who is experienced with other VR interfaces (including raycast-based selection), but who had previously only used the interfaces in the study in a limited form for testing purposes. Although with such a limited sample size, no valid statistical conclusions can be drawn from the results, we present metrics to facilitate discussion and analysis.

The 36 phrase set was divided equally at random between the interfaces so that each interface was tested with 2 practice phrases and 10 recorded phrases. Text entry studies usually use either blind trials where the participant can see that a character has been output but not the specific value, or perfect trials where the user is forced to correct errors before they can advance. The former preferences speed (since users do not have to spend time correcting errors), while the latter represents more realistic text entry scenarios. To provide balance, the first five recorded trials for each condition were

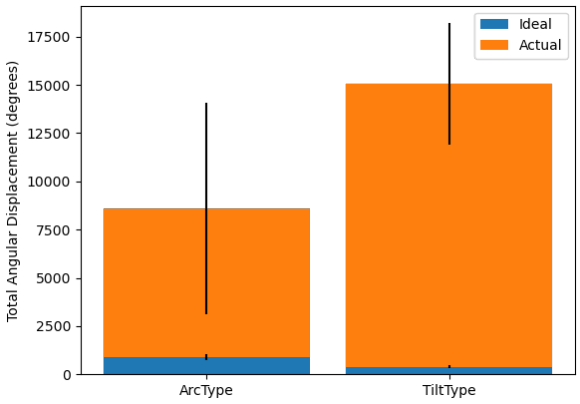

Figure 9: Total Angular Displacement. Bars indicate mean value, and the error bars are one standard deviation.

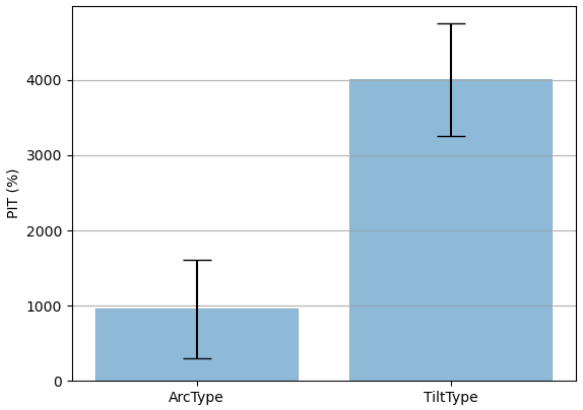

Figure 10: Percent Ideal Travel (PIT) results. Bars indicate mean value, and the error bars are one standard deviation.

blind and the last five required perfect output. The trials started with a button click and ended automatically when the correct number of characters had been entered for blind trials or a correct output was obtained for perfect trials.

Metrics for evaluation include entry rates, error rates, and measures of travel distance. All metrics are summarized in Table 1. In addition to these metrics, data was collected on the position of the stylus on character selection.

4.3.1 Words Per Minute (WPM). WPM is commonly used to measure the text entry rate. A word is defined as five characters including the space. Let O be the output string, $|O|$ represent the length of O in characters, and t be the amount of time in seconds that the user spent inputing O. As in Speicher et al. [50], WPM is calculated as:

$$WPM = \frac{(|O|-1)\text{ chars}}{5\text{ chars/word}} \times \frac{60\text{ secs/min}}{t\text{ secs}} = \frac{12(|O|-1)}{t}\frac{\text{words}}{\text{min}}$$

Shown in Figure 8, the Controller Pointing interface was the fastest with 8.8 (SD=0.88) WPM for blind trials and 6.6 (SD=1.1) for perfect trials, followed by Tilt-Type with 5.3 (SD=0.37) WPM for blind trials and 4.6 (SD=0.5) for perfect trials. Arc-Type performed the slowest with 2.8 (SD=0.2) WPM for blind trials and 1.6 (SD=0.54) WPM for perfect trials.

4.3.2 Percent Error Rate (PER). PER is calculated using the Levenshtein distance (LD) [35] between the prompt and the output string. In addition to the previously defined symbols, let P be the prompt string.

$$PER = \frac{100 \times LD(P,O)}{max(|P|,|O|)}$$

Because by definition perfect trials do not contain errors, only blind trials are used for analysis. Tilt-Type had the lowest percentage of errors with a mean PER of 8.6 (SD=4.6) percent, followed by Controller Pointing interface with 15.1 (SD=6.9). Lastly, Arc-Type had the highest percentage of errors with 23.7 (SD=8.0) percent.

4.3.3 Percent Ideal Travel (PIT). Because Tilt-Type and Arc-Type rely heavily on rotations of the stylus, we are interested in the extent that a user can efficiently select the precise angular change to move from one character selection to another. Percent Ideal Travel calculates the ratio between actual travel, T, and the ideal travel, I. Travel is measured by the total angular displacement of the stylus orientation, i.e. the amount a user rotates the stylus to transition between selecting each character in a phrase.

The ideal travel is calculated depending on the layout and interface type. For Tilt-Type, I takes into account the need to rotate the stylus to select a bin and tilt the stylus to select a character within the bin. Let dz be angular rotation necessary to shift bins, and dx be the angular change to travel between centers of adjacent keys in a bin. Ideal travel is given by:

$$I = \sum^{|P|-1} ||\Delta(O_i, O_{i+1}) \cdot (dx, dz)||$$

where Δ returns the displacement in keys between two characters in a layout (e.g. using the Tilt-Type layout shown in Figure 3, we see that $\Delta(C,L) = (2,1)$). For Arc-Type, I is calculated similarly to the above equation, but only takes into account the rotation necessary to change bins (because character selection within a bin is accomplished with the slider). Thus, PIT is calculated as:

$$PIT = \frac{100 \times T}{I}$$

The total angular displacement for Arc-Type and Tilt-Type compared with the ideal travel is shown in Figure 9. The PIT for both is shown in Figure 10. Tilt-Type had more than four times the PIT compared to Arc-Type (4000 [SD=750]% compared to 960 [SD=660]%). In practice, human input will always result in a PIT far greater than 100.

4.3.4 Stylus Positioning. The position of the stylus controller was captured for each output character. Positions are shown as a point cloud in Figure 11 left. The projections onto the xy and yz-planes are shown in the center and right image respectively. The UI and visual feedback were displayed in the xy-plane.

Figure 11 center shows that the Controller Pointing and Arc-Type interfaces have more dispersed distributions, corresponding with greater physical navigation to select characters. The Tilt-Type

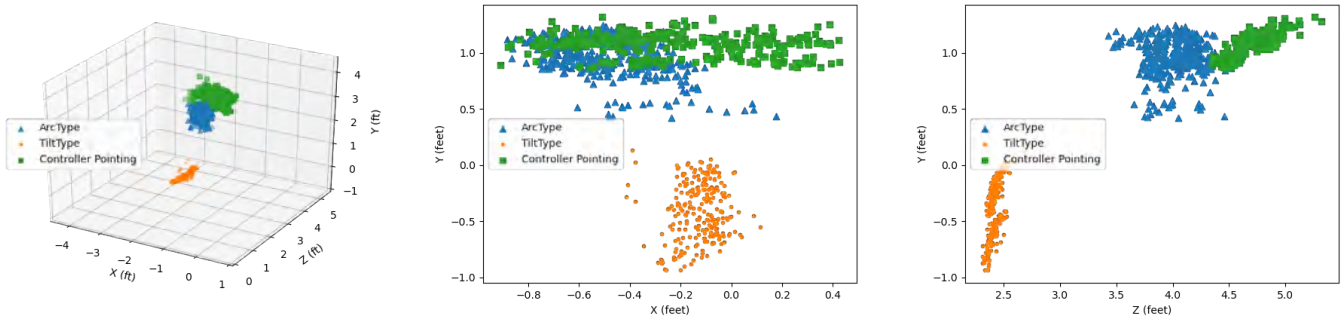

Figure 11: *Left:* Point clouds show the stylus position on each character selection. *Center:* Projection of the position data to the yx-plane (aligned with the UI). *Right:* Projection to the yz-plane.

interface shows a tighter distribution with less stylus translation. It is also lower in height corresponding with the ability to hold it in a relaxed posture at one's side rather than pointing at a key layout.

4.4 Discussion and Lessons Learned

The Controller Pointing interface had the fastest entry times but the second highest error rate compared to the other two interfaces. Its WPM rate is roughly half that reported in other studies using controller pointing (e.g. Boletsis and Kongsvik's 16.65 (3.28) WPM [5] and Speicher et al.'s 15.44 (2.68) [50]); however, both studies used bi-manual pointing with two controllers rather than one, which explains this doubling in speed.

The Controller Pointing interface's higher error rate may be attributed to several factors. Foremost is that small movements caused by fatigue or muscular jitter are amplified down the pointing ray, leading to larger variations of selection. Prior work has found that users typically have ±5 rotational degrees of noise when trying to hold a controller steady while pointing at a target [19]. This level of rotational noise can change the ray intersection point with a menu presented two meters from the user by almost 35 cm. This was less of an issue for Tilt-Type and Arc-Type because they only rely on angular changes at the origin of the stylus where 5 degrees is less than the rotation needed to change bins. We noticed that clicking the selection button while using the Controller Pointing interface introduced a strong downward deviation of the selection ray, previously coined as the "Heisenberg effect" [8]. In fact, 100 percent of the Controller Pointing interface errors were selections one letter below the target within the layout, compared with 11 percent in Tilt-Type, where the vertical orientation of the stylus prevented this issue. Over time, the tester found himself automatically compensating for this by making selections with the ray intersecting the top of a character box in the layout so that the downward movement would still be contained inside the key; a practice consistent with how users adapt to mis-recognition errors in gesture-based text entry interfaces [1]. It is possible that performance would be improved by modifying the bin layout to have greater height compared to width for Controller Pointing to account for these errors. This was not a problem for Arc-Type because it uses finger releases rather than button presses for confirmation.

It is useful to compare the jitter and Heisenberg effects for pen-based controllers with traditional VR wands. A study comparing interaction with laser pointers found that a wand-like device had less jitter than a pen-based laser pointer held in a precision grip [42]. Similarly, a VR study found that pen-based controllers had a higher level of high-frequency rotational jitter [3]. These results can be explained by basic physics. Wand controllers are usually large than pen-based ones, and the increased mass has a higher inertia requiring greater force to move them [2]. The use of a power grip rather than a precision grip also makes it easier to resist the forces of button-pressing. Several studies have explored grip forces on pens when writing [17] or drawing [24], but to our knowledge no evaluations of grip force when pressing buttons on pen-based controllers exist. In the precision grip (Shown in Figure 6) the thumb, index, and middle fingers form a small tripod that can act as a pivot point if a button is not directly aligned above the region where the fingers contact the stylus cylinder. This has important implications for the button layout when designing pen-based controllers. Despite the increased jitter, several recent studies (e.g. [3, 36, 45] have shown increased performance and fewer errors using pen-based controllers in a precision grip compared to traditional wands using power grips. This is likely because the precision grip provides increased dexterity and control, observed in children learning to write [48] and illustrated by the increased cortical area in the brain for processing finger motions compared with the wrist or arm [2, 62].

Compared to Controller Pointing, Tilt-Type's lower WPM may be attributed to the challenge of precisely selecting a particular angle. A prior study using a model based on Fitts' law found that controller pointing with a pen device is able to achieve a throughput of 4.7 bits/s. (Pointing with a standard controller only achieved 4.0 bits/s) [45]. In contrast, a study applying Fitts' law to tilt-based interaction with a mobile device found a throughput of 2.3 bits/s [39]. Our results are consistent with the lower throughput of angular selection. In a controlled study of pen-tilt interaction with surfaces, Xin et al. [58] found a decreasing power relationship between angular width and selection time. Tilt-Type's angular width to change between bins and characters is quite small to reduce the motion needed by the fingers. This likely led to additional rotation as the user first roughly targets the correct angle before slowing down to finely narrow in on the selection. This theoretical model of goal-directed aiming separates the movement into two phases: the initial ballistic phase and the correction phase [16], and it has been shown

to hold for 3D user interaction [14]. The PIT data support this conclusion, with the actual travel angle much higher than the ideal angle needed. The larger PIT for Tilt-Type compared to Arc-Type is likely due to this error being compounded for both the rotation and tilt angles in Tilt-Type compared with just using rotation for bin selection in Arc-Type, as well as the smaller angular widths in Tilt-Type . The error may also have been affected because jitter with smaller angular widths can lead to incorrectly changing bins on selection. 56 percent of errors with Tilt-Type were selections of the character in the bin to the left or right of the target. It is likely that Tilt-Type's entry speed and accuracy could be improved by increasing the possible range of motion (thus increasing angular widths) while having a trade-off in additional fatigue. Further study is needed to best characterize the ideal rotation ranges.

Arc-Type's high error percentage and low entry speed is likely due to a combination of the interface design and hardware. Although the finger release on the touch slider to confirm a selection avoids jittering the stylus, the physical movement of removing and replacing the finger is slower than maintaining a finger on a button. The hardware design may also have contributed to these issues. The smooth cylindrical barrel made it difficult to constantly maintain the same hand position on the stylus relative to the slider. This combined with limited haptic feedback made it harder to develop muscle memory for how far to extend the finger to select specific characters within a layout bin, leading to additional time after placing the finger at a position on the slider to move to the correct location. The increased errors are also likely due to the use of a potentiometer implementation for the slider. The potentiometer required medium to strong finger pressure. Not using enough pressure or slipping off of the edge of the slider while moving occasionally led to character selection errors. Although potentiometers are cheaper and more commonly available in the small size needed for the stylus design, we recommend the use of a capacitive touch sensor to improve the robustness for future pen-based controller designs, and care should be taken to avoid causing additional user stress with interaction techniques that use them.

In terms of comfort, Tilt-Type was strongly preferred for the small movements and ability to hold the stylus in a relaxed posture at the tester's side. Controller Pointing and Arc-Type both suffered from the so-called Gorilla arm, despite Arc-Type not depending on positional changes. Beyond ergonomics, the limited angular range for Tilt-Type and Arc-Type made it much easier to select characters along the borders of the layouts. Fully rotating to one side or the other maxed out the range without having to be precise. This was particularly useful because the space and backspace keys were mapped to the lower right corner of the layouts, making these commonly used keys the easiest to select. In contrast, the Controller Pointing interface requires precise selections for all the keys.

From this experience, we summarize our insights into the following lessons learned for pen-based text input design:

- Where entry speed is preferred, use Controller Pointing techniques, but account for button-press jitter errors that may be more prevalent compared to use of VR wand controllers. Button confirmations with a secondary controller held in the non-interacting hand may avoid issues.
- Favor angular-based approaches, like Tilt-Type, over positional ones to support the highest levels of comfort and avoid issues with button-press jitter.
- Pen-based text interfaces relying on rotation can leverage the easier selection of the outermost bins for commonly used characters (e.g. the space key).
- Physical buttons with haptic feedback lead to fewer errors compared with smooth slider input.
- Touch sliders on pen-like controllers should favor capacitive sensing rather than potentiometers.

5 LIMITATIONS AND FUTURE WORK

With restrictions due to Covid-19, the scope of the planned user study was severely restricted, making it challenging to generalize our results. It is possible that with one user, who had previous experience with raycasting-based interfaces, may have biased the results in favor of Controller Pointing. The lower number of users also made it impossible to counterbalance the interface order to avoid learning effects. In future work, we plan to implement a full controlled user study. If participants have prior experience with raycasting-based interfaces, this could potentially be balanced in the future with additional practice trials for Tilt-Type and Arc-Type. This will allow us to more fully characterize the trade-offs between these pen-based interfaces and more fully study the possible learning curve for Tilt-Type and Arc-Type.

In terms of improving performance, it is likely that both autocorrect and auto-complete algorithms commonly used on mobile devices would benefit Tilt-Type and Arc-Type, making them more widely useful. Although not used in the study, we adapted the SymSpell [23] symmetric delete search spelling correction algorithm and PruningRadixTrie [22] auto-completion library to provide these suggestions and auto-completion for a full interface. Further work is needed to identify how these features affect the accuracy and speed of the interaction.

Lastly, the alphabetical layouts were created to aid familiarity. The numbers in the alternate layouts are grouped to mimic a number pad on mobile phones. It is possible that the vertical progression of letters and the horizontal progression of numbers caused user confusion. Likely more optimized layouts exist based on letter frequency from bi-gram and tri-gram data. For example, the backspace key is so commonly used that placing it in the standard layout rather than the alternative layout would improve entry speed. Future work will entail exploring how these layouts interact with the angular selection options to optimize speed and accuracy, while using phrase sets that integrate punctuation and special characters (e.g. [54]).

6 CONCLUSION

In this paper, we presented two novel interfaces, Tilt-Type and Arc-Type , exploring the design space of text-input interfaces using a pen-shaped controller. These interfaces were designed to be immersive, fast, easy, and efficient for entering small snippets of text. Although they cannot match the speed of keyboard entry, they provide the critical ability to support text input at a room-scale without needing to return to a desk. Though initial evidence suggests that both interfaces are slower than Controller Pointing, Tilt-Type

was found to improve comfort and lower fatigue with fewer errors, while still maintaining an entry speed comparable with other room-scale text entry techniques. We hope that this work spurs further pen-based stylus interface development and explorations to improve the utility of room-scale VR/AR applications.

ACKNOWLEDGMENTS

This research is supported in part through a grant from the Macalester College Jr. Hub Faculty Research Fund, the Gordon Harrison Summer Research Collaborations Fund, and the Class of 1950 Summer Research Collaborations Fund. The software utilizes the VRPN library maintained by UNC-Chapel Hill's CISMM project with support from NIH/NCRR and NIH/NIBIB award 2P41EB002025.

REFERENCES

[1] Ahmed Sabbir Arif and Wolfgang Stuerzlinger. 2014. User adaptation to a faulty unistroke-based text entry technique by switching to an alternative gesture set. In *Proceedings of Graphics Interface 2014*. Canadian Human-Computer Communications Society, Toronto, Ontario, Canada, 183–192.

[2] Ravin Balakrishnan and I. Scott MacKenzie. 1997. Performance Differences in the Fingers, Wrist, and Forearm in Computer Input Control. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '97)*. Association for Computing Machinery, New York, NY, USA, 303–310. https://doi.org/10.1145/258549.258764

[3] Anil Batmaz, Aunnoy Mutasim, and Wolfgang Stuerzlinger. 2020. Precision vs. Power Grip: A Comparison of Pen Grip Styles for Selection in Virtual Reality. In *2020 IEEE Conference on Virtual Reality and 3D User Interfaces Abstracts and Workshops (VRW)*. IEEE Computing Society, New York, NY, USA, 23–28. https://doi.org/10.1109/VRW50115.2020.00012

[4] Leif P. Berg and Judy M. Vance. 2017. Industry use of virtual reality in product design and manufacturing: a survey. *Virtual Reality* 21, 1 (March 2017), 1–17. https://doi.org/10.1007/s10055-016-0293-9

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

[6] Costas Boletsis and Stian Kongsvik. 2019. Text Input in Virtual Reality: A Preliminary Evaluation of the Drum-Like VR Keyboard. *Technologies* 7, 2 (April 2019), 31. https://doi.org/10.3390/technologies7020031

[7] Sebastian Boring, Marko Jurmu, and Andreas Butz. 2009. Scroll, tilt or move it: using mobile phones to continuously control pointers on large public displays. In *Proceedings of the 21st Annual Conference of the Australian Computer-Human Interaction Special Interest Group: Design: Open 24/7 (OZCHI '09)*. Association for Computing Machinery, Melbourne, Australia, 161–168. https://doi.org/10.1145/1738826.1738853

[8] Doug Bowman, Chadwick Wingrave, Joshua Campbell, and Vinh Ly. 2001. Using pinch gloves for both natural and abstract interaction techniques in virtual environments. In *HCI International*. Springer, New Orleans, LA, USA, 629–633.

[9] Doug A. Bowman, Christopher J. Rhoton, and Marcio S. Pinho. 2016. Text Input Techniques for Immersive Virtual Environments: An Empirical Comparison:. *Proceedings of the Human Factors and Ergonomics Society Annual Meeting* 46, 26 (Nov. 2016), 2154–2158. https://doi.org/10.1177/154193120204602611 Publisher: SAGE PublicationsSage CA: Los Angeles, CA.

[10] Steven J. Castellucci, I. Scott MacKenzie, Mudit Misra, Laxmi Pandey, and Ahmed Sabbir Arif. 2019. TiltWriter: Design and Evaluation of a No-Touch Tilt-Based Text Entry Method for Handheld Devices. In *Proceedings of the 18th International Conference on Mobile and Ubiquitous Multimedia (MUM '19)*. Association for Computing Machinery, New York, NY, USA, Article 7, 8 pages. https://doi.org/10.1145/3365610.3365629

[11] Sibo Chen, Junce Wang, Santiago Guerra, Neha Mittal, and Soravis Prakkamakul. 2019. Exploring Word-gesture Text Entry Techniques in Virtual Reality. In *Extended Abstracts of the 2019 CHI Conference on Human Factors in Computing Systems (CHI EA '19)*. Association for Computing Machinery, Glasgow, Scotland Uk, 1–6. https://doi.org/10.1145/3290607.3312762

[12] Massless Corporation. 2020. Massless | Rethink and Redesign in 2D, 3D and VR. https://massless.io/massless-pen/ Library Catalog: massless.io.

[13] Samir Dash. 2017. BlueTap — The Ultimate Virtual-Reality (VR) Keyboard. https://medium.com/eunoia-i-o/bluetap-the-ultimate-virtual-reality-vr-keyboard-77f1e3d57d6f Library Catalog: medium.com.

[14] Cheng-Long Deng, Peng Geng, Yi-Fei Hu, and Shu-Guang Kuai. 2019. Beyond Fitts' Law: A Three-Phase Model Predicts Movement Time to Position an Object in an Immersive 3D Virtual Environment. *Human Factors* 61, 6 (2019), 879–894. https://doi.org/10.1177/0018720819831517

[15] Tafadzwa Joseph Dube and Ahmed Sabbir Arif. 2019. Text Entry in Virtual Reality: A Comprehensive Review of the Literature. In *Human-Computer Interaction. Recognition and Interaction Technologies (Lecture Notes in Computer Science)*, Masaaki Kurosu (Ed.). Springer International Publishing, Cham, 419–437. https://doi.org/10.1007/978-3-030-22643-5_33

[16] Digby Elliott, Werner F Helsen, and Romeo Chua. 2001. A century later: Woodworth's (1899) two-component model of goal-directed aiming. *Psychological bulletin* 127, 3 (2001), 342–357. https://doi.org/10.1037/0033-2909.127.3.342

[17] Tiago H. Falk, Cynthia Tam, Heidi Schwellnus, and Tom Chau. 2010. Grip Force Variability and Its Effects on Children's Handwriting Legibility, Form, and Strokes. *Journal of Biomechanical Engineering* 132, 11 (2010). https://doi.org/10.1115/1.4002611

[18] Jacqui Fashimpaur, Kenrick Kin, and Matt Longest. 2020. PinchType: Text Entry for Virtual and Augmented Reality Using Comfortable Thumb to Fingertip Pinches. In *Extended Abstracts of the 2020 CHI Conference on Human Factors in Computing Systems (CHI EA '20)*. Association for Computing Machinery, Honolulu, HI, USA, 1–7. https://doi.org/10.1145/3334480.3382888

[19] Andrew Forsberg, Kenneth Herndon, and Robert Zeleznik. 1996. Aperture based selection for immersive virtual environments. In *Proceedings of the 9th annual ACM symposium on User interface software and technology*. Association for Computing Machinery, New York, NY, USA, 95–96.

[20] S. Frees, R. Khouri, and G.D. Kessler. 2006. Connecting the Dots: Simple Text Input in Immersive Environments. In *IEEE Virtual Reality Conference (VR 2006)*. IEEE, Alexandria, VA, USA, 265–268. https://doi.org/10.1109/VR.2006.36

[21] Zhangjie Fu, Jiashuang Xu, Zhuangdi Zhu, Alex X. Liu, and Xingming Sun. 2019. Writing in the Air with WiFi Signals for Virtual Reality Devices. *IEEE Transactions on Mobile Computing* 18, 2 (Feb. 2019), 473–484. https://doi.org/10.1109/TMC.2018.2831709

[22] Wolf Garbe. 2020. wolfgarbe/PruningRadixTrie. https://github.com/wolfgarbe/PruningRadixTrie original-date: 2020-05-28T18:15:03Z.

[23] Wolf Garbe. 2020. wolfgarbe/SymSpell. https://github.com/wolfgarbe/SymSpell original-date: 2014-03-25T11:01:35Z.

[24] Arthur Gatouillat, Antoine Dumortier, Subashan Perera, Youakim Badr, Claudine Gehin, and Ervin Sejdić. 2017. Analysis of the pen pressure and grip force signal during basic drawing tasks: The timing and speed changes impact drawing characteristics. *Computers in Biology and Medicine* 87 (2017), 124–131. https://doi.org/10.1016/j.compbiomed.2017.05.020

[25] Jens Grubert, Lukas Witzani, Eyal Ofek, Michel Pahud, Matthias Kranz, and Per Ola Kristensson. 2018. Effects of Hand Representations for Typing in Virtual Reality. In *2018 IEEE Conference on Virtual Reality and 3D User Interfaces (VR)*. IEEE, Reutlingen, 151–158. https://doi.org/10.1109/VR.2018.8446250

[26] Jens Grubert, Lukas Witzani, Eyal Ofek, Michel Pahud, Matthias Kranz, and Per Ola Kristensson. 2018. Text Entry in Immersive Head-Mounted Display-Based Virtual Reality Using Standard Keyboards. In *2018 IEEE Conference on Virtual Reality and 3D User Interfaces (VR)*. IEEE, Reutlingen, 159–166. https://doi.org/10.1109/VR.2018.8446059

[27] NaturalPoint Inc. 2020. Motion Capture Systems. http://optitrack.com/index.html Library Catalog: optitrack.com.

[28] Bret Jackson. 2020. OVR Stylus: Designing Pen-Based 3D Input Devices for Virtual Reality. In *2020 IEEE Conference on Virtual Reality and 3D User Interfaces Abstracts and Workshops (VRW)*. IEEE, Virtual, 13–18. https://doi.org/10.1109/VRW50115.2020.00287

[29] Bret Jackson and Dan Keefe. 2019. MinVR/MinVRUnity. https://github.com/MinVR/MinVRUnity original-date: 2019-08-20T17:03:02Z.

[30] Haiyan Jiang and Dongdong Weng. 2020. HiPad: Text entry for Head-Mounted Displays Using Circular Touchpad. In *2020 IEEE Conference on Virtual Reality and 3D User Interfaces (VR)*. IEEE, Atlanta, GA, USA, 692–703. https://doi.org/10.1109/VR46266.2020.00092

[31] Haiyan Jiang, Dongdong Weng, Zhenliang Zhang, Yihua Bao, Yufei Jia, and Mengman Nie. 2018. HiKeyb: High-Efficiency Mixed Reality System for Text Entry. In *2018 IEEE International Symposium on Mixed and Augmented Reality Adjunct (ISMAR-Adjunct)*. IEEE, Munich, Germany, 132–137. https://doi.org/10.1109/ISMAR-Adjunct.2018.00051

[32] Janis Gisel Jimenez. 2017. *A Prototype for Text Input in Virtual Reality with a Swype-like Process Using a Hand-tracking Device*. Ph.D. Dissertation. UC San Diego. https://escholarship.org/uc/item/43r3t7m8

[33] Sooyoung Kim and Gerard Jounghyun Kim. 2004. Using keyboards with head mounted displays. In *Proceedings of the 2004 ACM SIGGRAPH international conference on Virtual Reality continuum and its applications in industry (VRCAI '04)*. Association for Computing Machinery, Singapore, 336–343. https://doi.org/10.1145/1044588.1044662

[34] Pascal Knierim, Valentin Schwind, Anna Maria Feit, Florian Nieuwenhuizen, and Niels Henze. 2018. Physical Keyboards in Virtual Reality: Analysis of Typing Performance and Effects of Avatar Hands. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems (CHI '18)*. Association for Computing Machinery, Montreal QC, Canada, 1–9. https://doi.org/10.1145/3173574.3173919

[35] V. I. Levenshtein. 1966. Binary Codes Capable of Correcting Deletions, Insertions and Reversals. *Soviet Physics Doklady* 10 (Feb. 1966), 707.

[36] Nianlong Li, Teng Han, Feng Tian, Jin Huang, Minghui Sun, Pourang Irani, and Jason Alexander. 2020. Get a Grip: Evaluating Grip Gestures for VR Input Using a Lightweight Pen. In *Proceedings of the 2020 SIGCHI Conference on Human Factors in Computing Systems (CHI '20)*. Association for Computing Machinery, New York, NY, USA, 1–13. https://doi.org/10.1145/3313831.3376698

[37] Logitech. 2020. Logitech VR Ink Pilot Edition - VR Stylus for Creativity, Control & Precision. https://www.logitech.com/en-us/promo/vr-ink.html Library Catalog: www.logitech.com.

[38] I. Scott MacKenzie and R. William Soukoreff. 2003. Phrase sets for evaluating text entry techniques. In *CHI '03 Extended Abstracts on Human Factors in Computing Systems (CHI EA '03)*. Association for Computing Machinery, Ft. Lauderdale, Florida, USA, 754–755. https://doi.org/10.1145/765891.765971

[39] I. Scott MacKenzie and Robert J. Teather. 2012. FittsTilt: The Application of Fitts' Law to Tilt-Based Interaction. In *Proceedings of the 7th Nordic Conference on Human-Computer Interaction: Making Sense Through Design* (Copenhagen, Denmark) *(NordiCHI '12)*. Association for Computing Machinery, New York, NY, USA, 568–577. https://doi.org/10.1145/2399016.2399103

[40] Bernard Marr. 2019. 5 Important Augmented And Virtual Reality Trends For 2019 Everyone Should Read. https://www.forbes.com/sites/bernardmarr/2019/01/14/5-important-augmented-and-virtual-reality-trends-for-2019-everyone-should-read/ Library Catalog: www.forbes.com Section: Innovation.

[41] Mark McGill, Daniel Boland, Roderick Murray-Smith, and Stephen Brewster. 2015. 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)*. Association for Computing Machinery, Seoul, Republic of Korea, 2143–2152. https://doi.org/10.1145/2702123.2702382

[42] Brad A. Myers, Rishi Bhatnagar, Jeffrey Nichols, Choon Hong Peck, Dave Kong, Robert Miller, and A. Chris Long. 2002. Interacting at a Distance: Measuring the Performance of Laser Pointers and Other Devices. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '02)*. Association for Computing Machinery, New York, NY, USA, 33–40. https://doi.org/10.1145/503376.503383

[43] PaleBlue Newsroom. 2020. Increased Use of VR in Everyday Industries. https://pale.blue/2020/03/30/increased-use-of-vr-in-everyday-industries/ Library Catalog: pale.blue Section: Blog.

[44] Duc-Minh Pham and Wolfgang Stuerzlinger. 2019. HawKEY: Efficient and Versatile Text Entry for Virtual Reality. In *25th ACM Symposium on Virtual Reality Software and Technology (VRST '19)*. Association for Computing Machinery, Parramatta, NSW, Australia, 1–11. https://doi.org/10.1145/3359996.3364265

[45] Duc-Minh Pham and Wolfgang Stuerzlinger. 2019. Is the Pen Mightier than the Controller? A Comparison of Input Devices for Selection in Virtual and Augmented Reality. In *25th ACM Symposium on Virtual Reality Software and Technology (VRST '19)*. Association for Computing Machinery, Parramatta, NSW, Australia, 1–11. https://doi.org/10.1145/3359996.3364264

[46] Sebastian Pick, Andrew S. Puika, and Torsten W. Kuhlen. 2016. SWIFTER: Design and evaluation of a speech-based text input metaphor for immersive virtual environments. In *2016 IEEE Symposium on 3D User Interfaces (3DUI)*. IEEE Computing Society, New York, NY, USA, 109–112.

[47] Ivan Poupyrev, Numada Tomokazu, and Suzanne Weghorst. 1998. Virtual Notepad: handwriting in immersive VR. In *Proceedings. IEEE 1998 Virtual Reality Annual International Symposium (Cat. No.98CB36180)*. IEEE Comput. Soc, Atlanta, GA, USA, 126–132. https://doi.org/10.1109/VRAIS.1998.658467

[48] Colleen M. Schneck and Anne Henderson. 1990. Descriptive analysis of the developmental progression of grip position for pencil and crayon control in nondysfunctional children. *The American journal of occupational therapy : official publication of the American Occupational Therapy Association* 44, 10 (1990), 893–900.

[49] Ben Shneiderman. 2000. The Limits of Speech Recognition. *Commun. ACM* 43, 9 (Sept. 2000), 63–65. https://doi.org/10.1145/348941.348990

[50] Marco Speicher, Anna Maria Feit, Pascal Ziegler, and Antonio Krüger. 2018. Selection-based Text Entry in Virtual Reality. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems (CHI '18)*. Association for Computing Machinery, Montreal QC, Canada, 1–13. https://doi.org/10.1145/3173574.3174221

[51] Russell M. Taylor, Thomas C. Hudson, Adam Seeger, Hans Weber, Jeffrey Juliano, and Aron T. Helser. 2001. VRPN: a device-independent, network-transparent VR peripheral system. In *Proceedings of the ACM symposium on Virtual reality software and technology (VRST '01)*. Association for Computing Machinery, Baniff, Alberta, Canada, 55–61. https://doi.org/10.1145/505008.505019

[52] Unity Technologies. 2020. Unity - Manual: Unity User Manual (2019.3). https://docs.unity3d.com/2019.3/Documentation/Manual/index.html Library Catalog: docs.unity3d.com.

[53] Keith Vertanen. 2009. *Efficient Correction Interfaces for Speech Recognition*. Ph.D. Dissertation. University of Cambridge, Cambridge, UK.

[54] Keith Vertanen and Per Ola Kristensson. 2011. A Versatile Dataset for Text Entry Evaluations Based on Genuine Mobile Emails. In *Proceedings of the 13th International Conference on Human Computer Interaction with Mobile Devices and Services (MobileHCI '11)*. Association for Computing Machinery, New York, NY, USA, 295–298. https://doi.org/10.1145/2037373.2037418

[55] Keith Vertanen, Haythem Memmi, Justin Emge, Shyam Reyal, and Per Ola Kristensson. 2015. VelociTap: Investigating Fast Mobile Text Entry using Sentence-Based Decoding of Touchscreen Keyboard Input. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems (CHI '15)*. Association for Computing Machinery, Seoul, Republic of Korea, 659–668. https://doi.org/10.1145/2702123.2702135

[56] James Walker, Bochao Li, Keith Vertanen, and Scott Kuhl. 2017. Efficient Typing on a Visually Occluded Physical Keyboard. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems (CHI '17)*. Association for Computing Machinery, Denver, Colorado, USA, 5457–5461. https://doi.org/10.1145/3025453.3025783

[57] Daniel Wigdor and Ravin Balakrishnan. 2003. TiltText: using tilt for text input to mobile phones. In *Proceedings of the 16th annual ACM symposium on User interface software and technology (UIST '03)*. Association for Computing Machinery, Vancouver, Canada, 81–90. https://doi.org/10.1145/964696.964705

[58] Yizhong Xin, Xiaojun Bi, and Xiangshi Ren. 2011. Acquiring and pointing: an empirical study of pen-tilt-based interaction. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '11)*. Association for Computing Machinery, Vancouver, BC, Canada, 849–858. https://doi.org/10.1145/1978942.1979066

[59] Wenge Xu, Hai-Ning Liang, Yuxuan Zhao, Tianyu Zhang, Difeng Yu, and Diego Monteiro. 2019. RingText: Dwell-free and hands-free Text Entry for Mobile Head-Mounted Displays using Head Motions. *IEEE Transactions on Visualization and Computer Graphics* 25, 5 (May 2019), 1991–2001. https://doi.org/10.1109/TVCG.2019.2898736

[60] Chun Yu, Yizheng Gu, Zhican Yang, Xin Yi, Hengliang Luo, and Yuanchun Shi. 2017. Tap, Dwell or Gesture? Exploring Head-Based Text Entry Techniques for HMDs. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems (CHI '17)*. Association for Computing Machinery, Denver, Colorado, USA, 4479–4488. https://doi.org/10.1145/3025453.3025964

[61] Difeng Yu, Kaixuan Fan, Heng Zhang, Diego Monteiro, Wenge Xu, and Hai-Ning Liang. 2018. PizzaText: Text Entry for Virtual Reality Systems Using Dual Thumbsticks. *IEEE Transactions on Visualization and Computer Graphics* 24, 11 (Nov. 2018), 2927–2935. https://doi.org/10.1109/TVCG.2018.2868581 Conference Name: IEEE Transactions on Visualization and Computer Graphics.

[62] Shumin Zhai, Paul Milgram, and William Buxton. 1996. The Influence of Muscle Groups on Performance of Multiple Degree-of-Freedom Input. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '96)*. Association for Computing Machinery, New York, NY, USA, 308–315. https://doi.org/10.1145/238386.238534