Matching gestures on word-gesture keyboards in VR with Bézier curves

Stiliyan Nanovski
Supervisor(s): Ricardo Marroquim, Amir Zaidi
EEMCS, Delft University of Technology, The Netherlands

A Thesis Submitted to EEMCS Faculty Delft University of Technology,
In Partial Fulfilment of the Requirements
For the Bachelor of Computer Science and Engineering
June 24, 2024

Name of the student: Stiliyan Nanovski
Final project course: CSE3000 Research Project
Thesis committee: Ricardo Marroquim, Amir Zaidi, Chirag Raman

Abstract

Effective text input in VR is an ongoing problem with no best-agreed-upon solution. One effective solution for this is a word-gesture keyboard implemented in VR. These keyboards work by matching the user-drawn gestures to predefined gestures of all words that connect consecutive letters of the word with straight lines. We investigate matching the user input with gestures comprising of Bézier curves. We found a small improvement in writing speed and a reduction in errors while writing. The method of using complex curves for matching gestures seems promising with many possible ways for further extension.

1 Introduction

Virtual Reality (VR) technology usage has significantly increased in recent years [1]. It has a wide range of uses, such as recreational purposes, video games, corporate meetings and brainstorming sessions. Text input is a core aspect of most human-computer interaction systems, including VR, making it worthwhile to try to improve.

Currently, there is no consensus among researchers for a single best way to input text, as many different methods show promising results [2, 3, 4, 5]. While inputting text through a physical keyboard is often the preferred way, this input method is not ideal, as having to remove one's VR headset is inconvenient and going in and out of the virtual environment hurts its immersion. Immersion should be preserved as much as possible, as it is a core part of VR technology. Methods entirely inside of the VR environment usually lack speed or put too much strain on the user's writing. This makes the question of finding the best text input method entirely in VR important.

The word-gesture keyboard is one such method that has shown promise. Word-gesture keyboards already exist outside of VR environments and past research [6, 7, 8] of their implementations in VR shows good performance. In this method, the user makes a swiping gesture through a keyboard layout by passing through the letters of the word they want to input. The gesture is then analysed and a ranking of the best matching words is produced. Figure 1 shows an example of how the gestures are drawn on a word-gesture keyboard.

Figure 1: Picture of the word-gesture keyboard in VR being used. The user is drawing the word "again" by drawing the gesture through the keys a-g-a-i-n.

One way to recognise gestures is by building a dictionary of what the gesture for each word would ideally look like. We refer to them as *perfect gesture graphs* (PGGs). In most implementations of word-gesture keyboards [9, 10] PGGs connect the centres of each key in a word with straight lines from one to the next. However, a person very often cannot draw completely straight lines, especially if their goal is to achieve a high writing speed.

In this paper, we modify the SHARK^2 [9] algorithm to match the input gestures to PGGs consisting of Cubic Bézier curves (Cubic Bézier PGGs) rather than straight lines (Line PGGs). We theorise that due to the movement in 3-Dimensional (3D) space and lack of friction from writing on a surface, a person's gesture when projected onto a 2D plane is more likely to resemble a curve than a straight line. We chose to use Bézier curves as they are very expressive using few parameters.

The rest of the paper is structured as follows: In Chapter 2, we go over related work for the word-gesture keyboards and alternative text input methods. Chapter 3 describes how Bézier curves were constructed and used, as well as explains the user study we conducted. The results of the study are shown in Chapter 4 and discussed in Chapter 5. We perform a user study comparing Line PGGs and Cubic Bézier PGGs, summarising the results, and discussing them. Chapter 6 goes over how user data was handled and provides a self-reflection on the research process. Finally, Chapter 7 is the conclusion of the paper and we discuss future work.

2 Related Work

In this section, we briefly examine other text input options for VR and how they compare to word-gesture keyboards. We also show existing research on word-gesture keyboards in VR and how they deal with PGGs.

2.1 Other text input methods

A lot other methods exist for text input in VR that show good performance in writing. Methods that only use a head-mounted display and controllers include raycast keyboards [11], drum-like keyboards [3] and speech-to-text [4]. A study performed by Boletsis and Kongsvik [2] compared all of these methods, as well as the word-gesture keyboard, and concluded that all of them perform well in VR. Multiple methods with specialised hardware for text input have been researched [12, 5] and overall perform better than methods with no special hardware, however, they are not easily available to most users due to lesser availability and higher monetary cost.

2.2 Word-gesture keyboards

The use of word-gesture keyboards in VR has seen significant research in the past [6, 7, 8]. There exist two notable variations of them in VR, differing by how the gestures are drawn: using head movement [7] and using controllers [6, 8]. Both methods have a good word-per-minute (WPM) performance, especially after multiple sessions or extended practice, lower error rate when typing, and a good score on the System Usability Scale [13], which is standard for usability testing [14]. We have opted to use the controller version, as it resembles physical word-gesture keyboards more closely, it performs better on usability score and there is a readily available open-source implementation [6]. The latter is important because it allows us to more directly compare our results to the previous study, as we do in Section 4.

For the recognition of gestures, existing research on word-gesture keyboards opts to implement the standard or simplified version of the SHARK^2 [9] or Vulture [10] algorithm. Both algorithms use PGGs with straight lines between consecutive keys. Shen et al. [15] created an algorithm specifically for recognising 3D gestures in VR. Their system shows improved performance over gestures mapped onto 2D.

3 Methodology

To attempt to improve upon the existing word-gesture keyboard, we use PGGs composed of Bézier curves instead of straight lines. The Bézier curves used are fit to writing data and then tested through a user study for their performance against the straight lines method.

3.1 Curve definition

We compare two different types of PGGs - straight lines and cubic Bézier curves. To create the PGG for a specific word we use a *template curve*, which falls into one of the two types. The template curve is a normalised curve that starts at the point $(0, 0)$ and ends at $(1, 0)$. For straight lines, this is the line connecting the points $(0, 0)$ and $(1, 0)$. The template Bézier curves have the first control point set to $(0, 0)$, the last to $(1, 0)$ and the middle control points can vary.

To create the PGG of a word w, we use the centres of the keys on the keyboard in the word, defined as $K_i \in \mathbb{R}^2$, $0 \leq i < len(w)$, where the points follow the same order as the letters in the word. For each pair of consecutive points (K_i, K_{i+1}), a template curve is scaled, rotated and translated, such that the points $(0, 0)$ and $(1, 0)$ correspond to K_i and K_{i+1} respectively. Figure 2 is an example of how to construct a PGG from a given template curve.

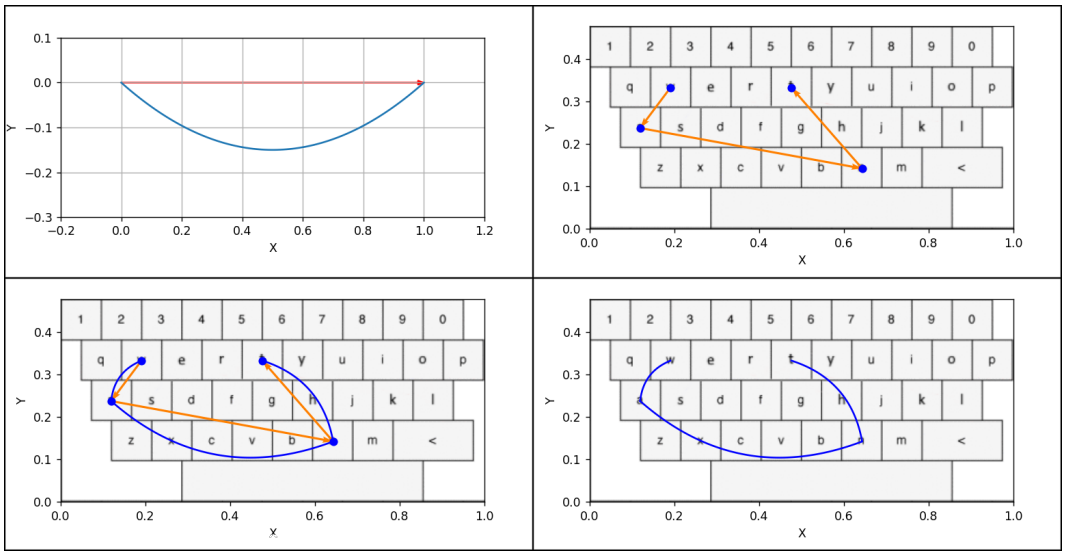

Figure 2: Example of constructing a PGG from a template curve and a word. The top left is the template curve, a quadratic Bézier curve picked only for illustration. The orange arrow shows orientation. The top right is the keys of the word "want", with the arrows showing the order of the letters. The bottom left is the oriented template curves placed to match the keys. The bottom right is the final PGG for this curve.

3.2 Fitting Bézier curves

Given a list of words and the user gestures that correspond to them, we want to find a template Cubic Bézier curve so that the produced PGGs fit as best as possible to the data. We do this by using the score function used by the SHARK^2 [9] and the open-source keyboard. For a gesture, the algorithm would produce a score for each word, then normalise the scores and order them, with higher scores indicating a better fit. To evaluate how well the gesture fits the PGG, we only need the score before it is normalised with the other scores. Instead, we only normalise it to be always between 0 and 1, relative to the maximum possible score. After calculating the score for each training word sample, we take the mean of all scores as the final evaluation of the template curve.

To optimise the control points of the template Bézier curve, we use the scoring function described above with basin-hopping to find an optimal value. Specifically, we used the method scipy.optimize.basinhopping from the SciPy Python library[1], using the default parameters and a starting value of $P_1 = (\frac{1}{3}, 0), P_2 = (\frac{2}{3}, 0)$ for the control points of the Cubic Bézier curve. We picked these points as they represent a line template curve so that its score is improved immediately. This way we arrived at the optimal template curve for the user study.

3.3 User Study

We performed a user study to compare the writing performance between the Line PGGs and Cubic Bézier PGGs. Participants were asked to write phrases from the MacKenzie phrase

[1]https://scipy.org

set [16] using the word-gesture keyboard. They were tasked to write each phrase exactly as given and to correct any mistakes they made while writing. They were not informed about the different PGGs used for the recognition of their gesture. Each participant had to write 16 phrases, which alternated between the two PGG types. None of the participants had previous experience in using word-gesture keyboards in VR and all wrote using their right hand.

Before the experiment, each participant was given up to 5 minutes to get accustomed to using the keyboard and to ask questions. They were explained how to write using gestures, how to use the autocomplete feature and how to erase words. They were told only the last word can be erased, so to correct a previous mistake they would have to also erase any words written after it. Afterwards, the user presses a button to begin the experiment and phrases are displayed one at a time for them to copy.

For each phrase, the metrics words-per-minute (WPM) and total error rate (TER) [17] are recorded. WPM is calculated from the moment the participants hold the trigger button to start writing their first word to the moment they release the trigger on the last word of the phrase, as long as it completes the phrase. We take the total non-space characters in the phrase into consideration and assume an average word length of 5. For TER, because all mistakes are required to be corrected, only the number of words *incorrect fixed* (IF) mistakes are relevant. Both metrics are analysed per user and not per phrase, by taking the mean WPM and TER per user.

We used an existing, open-source implementation of a word-gesture keyboard in VR[2], the one used by Spiess et. al. [6]. Each trial was performed with the same hardware: A Quest 2 M2 Plus VR headset connected to a computer with an AMD Ryzen 7 2700X CPU and an NVIDIA GeForce RTX 2080 GPU.

The user study was done in two parts, only differing in the parameters of the Bézier curve PGGs used. The first part used curves that were fit to the data of one person repeatedly using the keyboard, generating 100 phrases. The second part used curves that were fit to the data collected from the first part of the study. The fitting was done as prescribed in Section 3.2. The Desmos[3] visualisation in Figure 3 shows the template curves used in each part of the user study.

[2]https://github.com/Philipp1202/WGKeyboard
[3]https://www.desmos.com/calculator

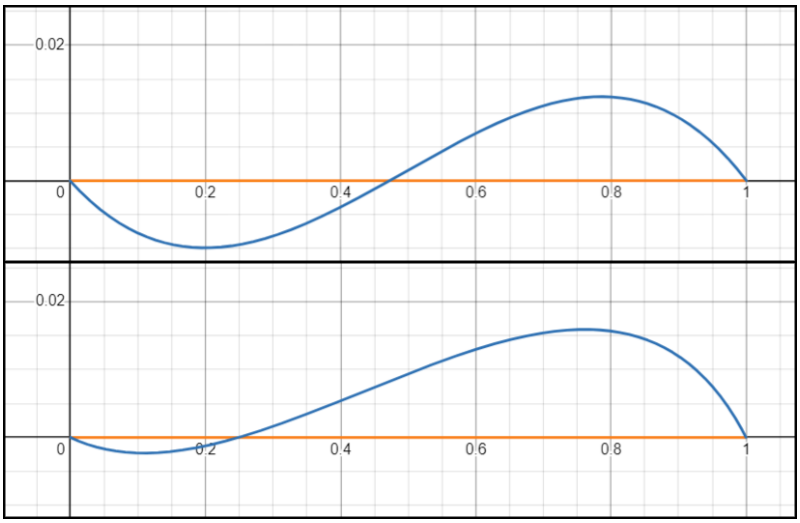

Figure 3: Template curves used in part 1 (top) and part 2 (bottom) of the user study. The coordinates of the middle control points are $P_1 = (0.326, -0.036), P_2 = (0.688, 0.041)$ for part 1 and $P_1 = (0.291, -0.014), P_2 = (0.785, 0.042)$ for part 2. The visualisations were created using Desmos.

4 Results

A total of 13 people participated in the first part of the user study and 9 in the second part. Two people from the first part experienced technical difficulties during their trials. For the first person, their VR environment crashed during the experiment. The environment was restarted and they were allowed to continue writing until they wrote 16 phrases collectively, however, they wrote more phrases using Line PGGs than with Cubic Bézier PGGs. For the second one, after the experiment, their data was incorrectly saved and ultimately lost, leading to 21 people's data in total.

Initially, we collected 192 phrases from the first part of the survey and 144 from the second. Because the users were required to fix all errors in their writing but were unable to change a single word back freely, many phrases were written significantly slower than they would be otherwise. For this reason, we limited the data only to phrases that did not require the erasure of correctly written words. After this, the data is reduced to 122 phrases in part one and 91 in part two.

After fitting the Cubic Bézier PGGs to the data obtained from part 1, they perform slightly better on both metrics than Line PGGs. The difference in the metrics is not statistically significant for WPM ($p \approx 0.426, t \approx 0.808$) or for TER ($p \approx 0.534, t \approx 0.630$). Our results do not significantly differ from the ones obtained by Spiess et. al. [6] for WPM ($p \approx 0.109, t \approx 1.659$), but they do for TER ($p < 0.01, t \approx 2.817$). Table 1 summarises the performance analysis of the user study.

Curve	WPM	TER	# Users	# Phrases
Line (our, parts 1 & 2)	11.41 *(2.15)*	18.62% *(8.63%)*	21	112
Line [6]	12.94 *(2.53)*	9.33% *(5.50%)*	8	120
Cubic Bézier (part 1)	10.62 *(2.41)*	23.99% *(9.62%)*	12	53
Cubic Bézier (part 2)	12.03 *(1.19)*	16.53% *(7.54%)*	9	46

Table 1: Results from both parts of the user study and the results of [6] on performance. The first row shows our combined results of using Line PGGs from both parts of the user study. The second row shows the results from [6] using Line PGGs. The last two rows are the results from using Cubic Bézier PGGs from parts 1 and 2 of the user study. WPM and TER are formatted as "mean *(std)*".

The results from fitting the Cubic Bézier PGGs to the data from part 1 are shown in Table 2 using the score described in Section 3.2. The fit Cubic Bézier PGGs show a significantly better score than Line PGGs ($p < 0.01, t \approx 2.994$), with an increase of the mean score by 7.6%.

Curve	Score	Words	Optimal Parameters
Line	0.5702 *(0.2536)*	604	N/A
Cubic Bézier	0.6136 *(0.2501)*	604	$P_1 = (0.291, -0.014), P_2 = (0.785, 0.042)$

Table 2: Score and optimal parameters of each curve type for the used training data. A total of 604 words were used while fitting the data, each with its score calculated as described in Section 3.2. The formatting used is "mean *(std)*". The optimal parameters show the intermediate control points of the template Cubic Bézier Curve, where the first and last control points are $(0, 0)$ and $(1, 0)$ respectively.

5 Discussion

The use of Cubic Bézier PGGs has shown a small improvement in writing performance over Line PGGs. We believe this is because the best-fitting cubic Bézier curves are very close shape-wise to straight lines. The line template curve has length 1, while the Cubic Bézier template differs from it by less than 0.02 at all points, as seen in Figure 4. The small difference between the curves suggests that in VR, users still succeed in drawing gestures close to straight lines with small variations.

An interesting observation of the fit Cubic Bézier PGGs is that its shape is asymmetric, where the first and second half of it are significantly different in size. This can be seen in the Desmos visualisation of the template curve shown in Figure

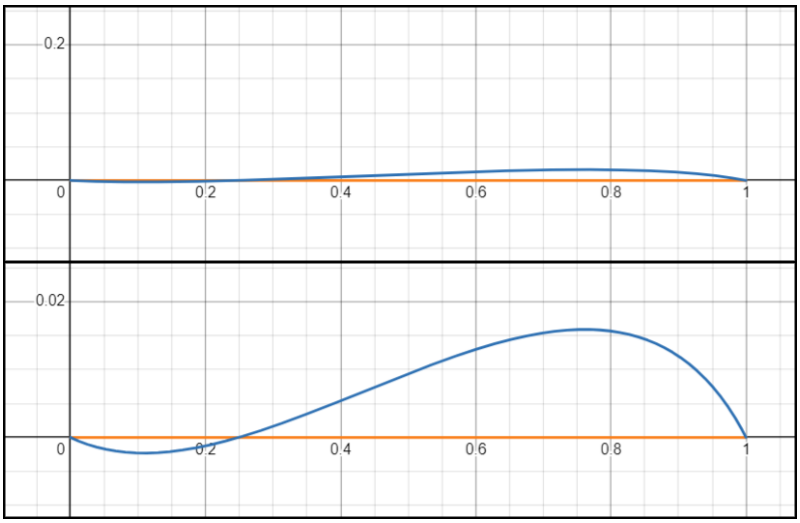

Figure 4: Continuous curve of the fit template curve for Cubic Bézier PGGs. The orange line and the blue curve are the template curves for Line PGGs and fit Cubic Bézier PGGs respectively. The top graphs display the axis in $1:1$ scale and the bottom displays the axis in $10:1$ scale. The visualisations were created using Desmos.

This shows that on average, when going from one key to the next, the end of the section is significantly further away from a straight line. This trend stays consistent when fitting the data to only the first part of the user study, fitting to both parts together and fitting to only the last 8 phrases of each user. This implies that even after getting experience in using the keyboard, users continue to make bigger arcs on the second part of the template curve. We believe this is due to the initial momentum from the first key being more accurate, then curving away from a straight line and completing the arc.

Because all participants used their right hand for writing, the Cubic Bézier PGGs were fit specifically for that. It is likely that had all participants been left-handed the final fit Cubic Bézier PGGs would look different, possibly a mirrored version of the current results. If data from both left- and right-handed gestures the fit PGGs might be even closer to the Line PGGs.

One significant difference between our results and those of Spiess et. al. [6] is in the TER metric. We believe this is most likely a difference in how the user studies were set up. This can either be because of a difference in participants' experience with VR or word-gesture keyboards, or with the directions the participants were given and what to focus on. Another possibility is in the way TER was calculated in the other study, as they do not explicitly state how it was calculated. Our TER is almost exactly twice as high as the previous study, which may suggest a different scale for calculation.

6 Responsible Research

During the research of the paper, special attention was given to ensuring that all performed research was conducted ethically and responsibly.

Data was collected from users through the user survey for analysis. To ensure the participants' data in the user study was properly handled before the study was conducted,

an application to the Human Research Ethics Committee of TU Delft was submitted and later approved for the study's setup. The application described what human data would be gathered and how it would be stored safely and anonymously. Furthermore, it assesses the risks of handling the data and how to take measures against these risks. All other data was gathered from publicly available open-source projects, licensed to allow usage and modification.

To allow the reproducibility of this research, all software and data generated throughout the Research Project course are publicly available. The software is released open-source on GitHub[4] and the user-generated data is stored on the 4TU Data Repository[5] and the TU Delft network[6]. The repository contains a "README" file with further instructions on how to run the code. The paper provides the exact specifications of the HMD, Controllers and Computer hardware used for the user study. Although the results do not correlate with the hardware used, they serve as an example of sufficient specifications. Using similar or better-performing hardware should not impact the results of the study.

7 Conclusion and Future Work

In this paper, we presented an improvement for word-gesture keyboards used in VR using the SHARK^2 algorithm. Cubic Bézier curves were used to get better accuracy and a lower error rate when recognising the gestures on the keyboard. A significant amount of data was collected for the use of the word-gesture keyboard which is publicly available for future analysis.

The idea of non-Line PGGs can be extended in multiple different ways. Higher-order Bézier PGGs should produce better results than Cubic Bézier PGGs, but the improvement might be less significant than the improvement from lines to cubic Bézier. Another change can be made to the orientation of the template curve. Our PGGs used the same orientation of the template curve, regardless of the positions of the keys. It is possible that changing the orientation for different letter transitions, by looking at the expected direction changes, can lead to a smoother curve and potentially better performance. Another way to fit the curves would be specifically for each user separately.

All data collected was from writing using the right hand and then fitting the Cubic Bézier PGGs to right-hand gestures. The left-hand gestures would likely produce different PGGs, possibly symmetrical or reflected to the right-hand ones.

Overall, the method we are proposing seems promising to improve word-gesture keyboards in VR, with a lot of possible improvements and research that can be done on the topic. The method described can also be applied to regular word-gesture keyboards and can more easily be tested on a larger scale due to not requiring VR equipment.

References

[1] Ayah Hamad and Bochen Jia. How virtual reality technology has changed our lives: an overview of the current and potential applications and limitations. *International journal of environmental research and public health*, 19(18):11278, 2022.

[4] https://github.com/snanovski/bezier-word-gesture-keyboard

[5] https://data.4tu.nl/private$_{d}atasets/gzu-mgxP87EqeBaXMnZHWiDqjyNmNf5h62ss1ehLthw$

[6] https://webdata.tudelft.nl/staff-umbrella/collabprojvr/, accessible through EduVPN

[2] Costas Boletsis and Stian Kongsvik. Controller-based text-input techniques for virtual reality: An empirical comparison. 2019.

[3] Costas Boletsis and Stian Kongsvik. Text input in virtual reality: A preliminary evaluation of the drum-like vr keyboard. *Technologies*, 7(2):31, 2019.

[4] Jiban Adhikary and Keith Vertanen. Text entry in virtual environments using speech and a midair keyboard. *IEEE Transactions on Visualization and Computer Graphics*, 27(5):2648–2658, May 2021.

[5] John Dudley, Hrvoje Benko, Daniel Wigdor, and Per Ola Kristensson. Performance envelopes of virtual keyboard text input strategies in virtual reality. In *2019 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, pages 289–300, Oct 2019.

[6] Florian Spiess, Philipp Weber, and Heiko Schuldt. Direct interaction word-gesture text input in virtual reality. In *2022 IEEE International Conference on Artificial Intelligence and Virtual Reality (AIVR)*, pages 140–143, Dec 2022.

[7] Chun Yu, Yizheng Gu, Zhican Yang, Xin Yi, Hengliang Luo, and Yuanchun Shi. 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*, pages 4479–4488, 2017.

[8] Sibo Chen, Junce Wang, Santiago Guerra, Neha Mittal, and Soravis Prakkamakul. Exploring word-gesture text entry techniques in virtual reality. In *Extended Abstracts of the 2019 CHI Conference on Human Factors in Computing Systems*, pages 1–6, 2019.

[9] Per-Ola Kristensson and Shumin Zhai. Shark2: a large vocabulary shorthand writing system for pen-based computers. In *Proceedings of the 17th Annual ACM Symposium on User Interface Software and Technology*, UIST '04, page 43â52, New York, NY, USA, 2004. Association for Computing Machinery.

[10] Anders Markussen, Mikkel Rønne Jakobsen, and Kasper Hornbæk. Vulture: a mid-air word-gesture keyboard. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 1073–1082, 2014.

[11] Yongjae Lee and Gerard J Kim. Vitty: Virtual touch typing interface with added finger buttons. In *Virtual, Augmented and Mixed Reality: 9th International Conference, VAMR 2017, Held as Part of HCI International 2017, Vancouver, BC, Canada, July 9-14, 2017, Proceedings 9*, pages 111–119. Springer, 2017.

[12] I. Poupyrev, N. Tomokazu, and S. Weghorst. Virtual notepad: handwriting in immersive vr. In *Proceedings. IEEE 1998 Virtual Reality Annual International Symposium (Cat. No.98CB36180)*, pages 126–132, March 1998.

[13] John Brooke et al. Sus-a quick and dirty usability scale. *Usability evaluation in industry*, 189(194):4–7, 1996.

[14] Philip T. Kortum Aaron Bangor and James T. Miller. An empirical evaluation of the system usability scale. *International Journal of HumanâComputer Interaction*, 24(6):574–594, 2008.

[15] Junxiao Shen, John Dudley, and Per Ola Kristensson. Fast and robust mid-air gesture typing for ar headsets using 3d trajectory decoding. *IEEE Transactions on Visualization and Computer Graphics*, 2023.

[16] I Scott MacKenzie and R William Soukoreff. Phrase sets for evaluating text entry techniques. In *CHI'03 extended abstracts on Human factors in computing systems*, pages 754–755, 2003.

[17] R William Soukoreff and I Scott MacKenzie. Metrics for text entry research: An evaluation of msd and kspc, and a new unified error metric. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 113–120, 2003.