Efficient Special Character Entry on a Virtual Keyboard by Hand Gesture-Based Mode Switching

Zhaomou Song* John J. Dudley† Per Ola Kristensson‡

University of Cambridge

ABSTRACT

The need to support efficient text input in virtual reality continues to attract significant research attention. However, much of this research understandably focuses exclusively on core text input tasks involving the entry of standard alphabetic characters. The less common though still critical task of entering special characters is often ignored. In this paper we focus on this niche use case as chiefly encountered when entering passwords. Current commercial virtual keyboards allow users to switch between different layers of the keyboard in order to access capital letters, numerals and special characters by pressing an explicit mode-switch button. We propose a new method of switching between layers of a virtual keyboard using hand gestures. Critically, these hand gestures are seamlessly performed in conjunction with key selections to deliver an efficient and intuitive interaction. We report on a user study with 16 participants entering standard passwords comparing our gesture-based mode switching approach to a conventional button-based baseline. We find that with practice our proposed method results in significantly faster entry rates without any deterioration in accuracy. Feedback from users also indicates that our technique is considered efficient and comfortable to use.

Index Terms: Human-centered computing—Human computer interaction (HCI)—Interaction techniques—Text input Human-centered computing—Human computer interaction (HCI)—Interaction paradigms—Virtual reality

1 INTRODUCTION

The Qwerty keyboard remains the default method for text input on most modern computing devices and there is a significant body of research examining how the Qwerty keyboard can be effectively adapted to different interaction settings. A mid-air virtual Qwerty keyboard for use in augmented reality (AR) or virtual reality (VR) can be readily implemented by virtualizing the concept of a touch. This means collisions between a virtual plane and the user's fingertip can be detected and treated in a similar way to physical touches on a capacitive touchscreen.

The virtualization of touches has been shown to work well in AR and VR (e.g. [6, 7]) and offers satisfactory entry rates for standard text input. However, the vast majority of text entry research in immersive settings focuses almost exclusively on entry of standard alphabetic characters. This is entirely reasonable given that entry of standard alphabetic characters is the most common use case for text input. Nevertheless, a comprehensive text input system should 'walk the last mile' [14] and must ultimately support entry of a complete set of numerals, symbols and punctuation to address less-common ancillary use cases, such as password entry or even entry of code syntax. In this paper we investigate this largely unaddressed requirement for extended character input on a mid-air virtual Qwerty keyboard.

Specifically, we examine the following research question: *How can we support efficient special character input on a mid-air virtual Qwerty keyboard?* An obvious solution, and the method used in modern AR and VR HMDs (e.g. Meta Quest 2 and Microsoft HoloLens 2), is to provide alternative keyboard layers that can be toggled by pressing a dedicated button on the keyboard. Indeed, this is a relatively straightforward transplantation of the method widely used on smartphone virtual keyboards (see e.g. Figure 1). A detriment of this approach is that a minimum of two key presses is required to access a secondary layer and a minimum of three key presses is required to access a tertiary layer when the user starts from the primary layer. An elementary analysis (i.e. the keystroke-level model [2]) tells us that any additional key presses required as part of an input process will incur an additional input time cost that may ultimately negatively impact efficiency.

Another widely applied and complementary strategy (supported on Gboard, and the system keyboards of Meta Quest 2 and Microsoft HoloLens 2) is overloading the primary layer with a subtle visualization of the secondary layer. This secondary layer can then be activated by means of a dwell interaction. This approach eliminates the need for an additional key press but with limited or no performance gain given the requirement for the dwell time.

Both conventional strategies for special character layer switching in AR or VR show limited evolution from their conceptual ancestors on a smartphone. This observation prompted our investigation into how the additional capabilities afforded by modern AR and VR HMDs, such as hand tracking-based gesture interaction, could be exploited to better streamline the process of entering special characters on a virtual Qwerty keyboard. To this end we propose and investigate the performance of a gesture-based layer switching method that is seamlessly integrated within a standard virtual touch-based interaction paradigm. This technique allows the user to perform touches on the keyboard while controlling the active keyboard layer using hand-gesture correspondence. Our evaluation shows that this proposed technique allows users to enter special characters more efficiently and is overwhelmingly preferred by the participants in our study compared to a commonly used baseline.

In summary, this paper offers the following two main contributions:

- A gesture-based technique for seamless keyboard layer switching to support efficient special character entry.
- An empirical evaluation comparing the performance of the gesture-based technique against a button-based layer switching baseline, in which we found a significant 8.3% and 21.8% increase in entry rate respectively for two password entry tasks.

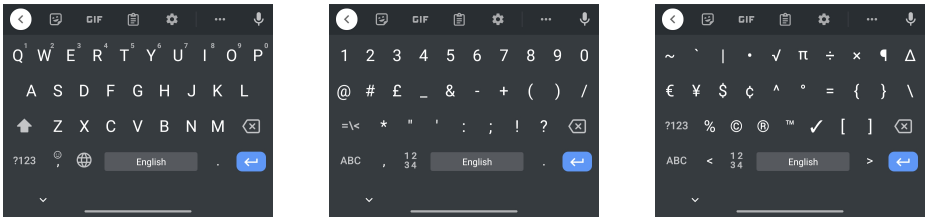
2 RELATED WORK

Mid-air virtual keyboards continue to attract active research attention. A multitude of interaction techniques and keyboard designs have been proposed and investigated (see Dube and Arif [5] for a comprehensive review of text entry in VR). Widely explored interaction methods for mid-air virtual Qwerty keyboards include controller

*e-mail: zs323@cam.ac.uk

†e-mail: jjd50@cam.ac.uk

‡e-mail: pok21@cam.ac.uk



(a) Alphabetic character layer (shown capitalized).

(b) Secondary layer with punctuation and numerals.

(c) Tertiary layer with symbols and punctuation.

Figure 1: The three main alternative character input layers of Google's Gboard that provided the basis for the chosen character layout.

and hand-based input [6, 7, 22], gaze or head-based input [15, 26] and novel device-based input [12]. Other efforts have focused on improving the design of a virtual keyboard by optimizing its location and orientation [10, 24]. Research has also examined different methods for improving entry rates, accuracy and user experience such as by integrating speech recognition [1] and tactile feedback [13]. McGill et al. [16] introduce a toolkit for creating virtual keyboards and augmenting physical keyboards in VR.

An illustrative example of the broad design space for mid-air text input in VR is the investigation by Speicher et al. [22]. Speicher et al. [22] investigated six alternative selection-based Qwerty keyboard input methods for use in VR: head pointing; controller pointing; controller tapping; freehand; discrete controller cursor and continuous controller cursor. The general recommendation from this paper was to use controller pointing where feasible. However, a controller can encumber the user and, in instances where the user is not already holding a controller when seeking to input text, it can be highly disruptive to require the user to locate and pick up a controller. There is therefore a clear need for controller-free text input methods and indeed the native hand-tracking capability of modern HMDs has improved significantly since the study by Speicher et al. [22]. Dudley et al. [7] investigated the performance potential of mid-air controller-free text input using tracked index fingers and found that entry rates of 30 to 50 words-per-minute (WPM) are feasible. Entry rates in the context of special character input for passwords are unlikely to reach this level given that users do not typically have a strong memory of where, or on what layer, a desired non-alphabetic character is located. Effective entry of passwords also demands that the user takes care to avoid input errors and this typically leads to lower entry rates. Nevertheless, users are still likely to benefit from methods that streamline the process of transitioning between layers and afford accurate selection of characters within a given layer. To the best of our knowledge, there is no prior work that specifically focuses on improving the performance of special character entry on a mid-air virtual keyboard.

As noted by Dube and Arif [5], "most existing text entry techniques for VR are adaptations of techniques that were designed and optimized for different form factors." This is particularly true for special character entry where current implementations are straightforward adaptions of the approach taken on smartphone keyboards. There has been limited work seeking to fully exploit the additional input modalities and interaction space afforded by immersive HMDs to improve the process of entering special characters. Nevertheless, there is some conceptually relevant work that illustrates the more general concept of leveraging additional information embedded in text input interaction to improve performance. For example, Weir et al. [25] measured touch pressure during typing on a physical input surface to implicitly model touch uncertainty. Foy et al. [9] similarly exploited an awareness of the finger used and dynamic characteristics of the touch to probabilistically model the likelihood of a spurious touch when typing on a mid-air VR keyboard. Both of these examples illustrate how additional information already captured by the system in text input interaction can be exploited to improve performance. Our gesture-based mode switching technique extends this general concept by enabling the user to actively trigger interface changes without interrupting the standard method of interacting with the keyboard.

Prior work has also looked at mode-switching of VR user interfaces and interactions within different application contexts. Surale et al. [23] evaluated a range of different actions, including some static hand gestures, as a means to perform generic mode switching in VR. Hand gesture-based interaction and mode selection is an attractive choice for VR and AR applications given the growing access to accurate hand tracking data and tools that streamline the creation of gesture recognizers [17, 20]. To distinguish between different possible actions resulting from the same gesture, Chen et al. [3] explored secondary input for disambiguation by means of head gaze, speech or foot taps. Pfeuffer et al. [18] investigated how gaze can be integrated with controller input to streamline menu interactions in a way that minimizes disruption of the main task. Shi et al. [21] evaluated the use of deliberate head motions sensed by an HMD as a means to toggle between tools in a controller-based 3D painting application. In this paper, we explore gesture-based mode switching for the specific task of transitioning between layers on a virtual keyboard.

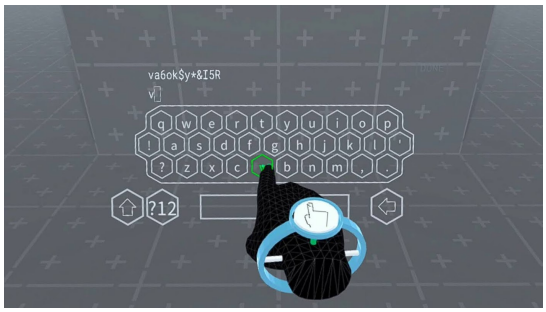
While there are some use cases where special characters are used in communication (e.g. emojis or mathematical operations), we consider authentication to be one of the most useful applications of gesture-based mode switching. Schneider et al. [19] introduced variety to the physical keyboard interface to support creative use cases in VR, one of which suggested shuffled keys to prevent shoulder-surfing during authentication. Other previous works aim to address the limitation of a text-based password by focusing on alternative methods for authentication in VR. George et al. [11] suggested authentication by connecting 3D objects in the virtual space as a method to leverage all three dimensions. Evtimova and Nicholson [8] proposed graphical passwords based on 2D images for users who have difficulty reading and memorizing text. There has been little research attention given to improving the typing experience of text-based passwords, despite the fact that this still remains the most common method of authentication.

3 System Design

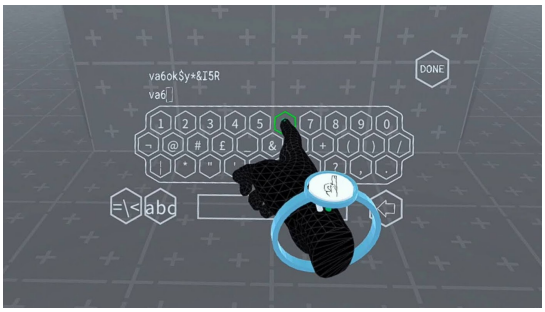
There are three main components of the gesture-based mode switching method: 1) the keyboard design; 2) the design of the gestures themselves; and 3) the method for performing rapid and accurate gesture recognition.

3.1 Keyboard Design

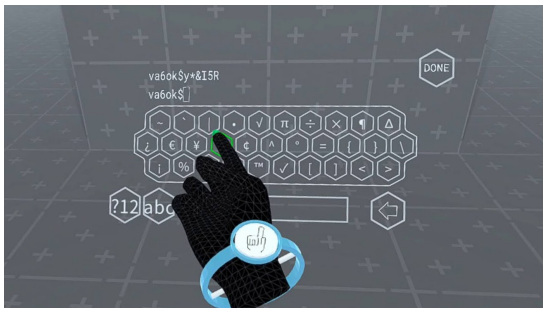
The keyboard presented in Dudley et al. [7] was extended to include secondary and tertiary input layers with numerals, symbols and punctuation. The user can switch between these layers using either our gesture-based switching method or a standard button-based interaction. The interaction design of the button-based switching



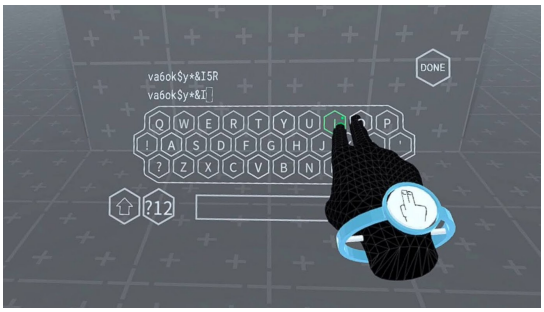
(a) INDEXPOINT-0: Typing on the primary alphabetic character layer.



(b) INDEXPOINT-90: Typing on the secondary layer with numerals.



(c) INDEXPOINT-180: Typing on the tertiary symbol and punctuation layer.



(d) DOUBLEPOINT: Typing on the capitalized alphabetic character layer.

Figure 2: The four hand gestures used to switch between keyboard input layers.

method replicates the behavior of Gboard. The ⟨*SHIFT*⟩ key toggles the capitalization of the primary layer. As illustrated in Figure 1, the secondary input layer is accessed by pressing the ⟨*?123*⟩ key shown in the bottom left of Figure 1a. After switching to the secondary input layer, the ⟨*SHIFT*⟩ is replaced by the ⟨=\<⟩ key as shown in Figure 1b. Pressing the ⟨=\<⟩ key brings up the tertiary input layer shown in Figure 1c. The tertiary layer contains the least commonly used special characters, hence the requirement for an additional keystroke to activate it. The ⟨*?123*⟩ key on the primary layer is replaced by an ⟨*ABC*⟩ key on the secondary and tertiary layer, allowing the user to switch back to the primary layer.

The behavior of these mode switching keys was replicated in our mid-air VR keyboard implementation to establish the button-based baseline method subsequently evaluated. We also adopted Gboard's allocation of numerals and punctuation to the secondary and tertiary layers, although the VR keyboard provides some additional key slots which were filled with additional symbols (see Figure 2).

3.2 Gesture Design

The gesture-based switching method triggered layer changes in the same way as the button-based approach. The frequency of use and associated meaning of these different layers is reflected in the design of these gestures.

We designed four gestures providing a one-to-one correspondence with the four keyboard layers: lowercase alphabetical characters; uppercase alphabetical characters; secondary numeral, symbol and punctuation layer; and tertiary symbol and punctuation layer. The philosophy behind the design of these gestures is to provide a method that can reduce the time and effort needed for layer switching during typing, while minimizing any disruption to the flow while typing. Figure 2 illustrates each gesture, as well as the layer it activates on the virtual keyboard. The four gestures are also summarized below:

1. INDEXPOINT-0: Typing with right index finger while palm is facing downwards. This gesture corresponds to the default alphabetic layer.

2. INDEXPOINT-90: Typing with right index finger while palm is facing to the left. This corresponds to the secondary layer with the punctuation, numerals and some special characters.

3. INDEXPOINT-180: Typing with right index finger while palm is facing upwards. This corresponds to the tertiary layer with punctuation and additional special characters.

4. DOUBLEPOINT: Typing with both right index finger and middle finger, while palm is facing downwards. This corresponds to the capitalized alphabetical character layer.

Each gesture can be performed with small adjustments of the hand while maintaining a pointing-forward form so that the typing process is minimally affected. Significantly, the transition between the primary, secondary and tertiary layers requires changing the hand posture in one axis only, that is, by rotating the wrist. This provides the user with a potential mental association between rotation degree and layer number. No wrist rotation (palm down) is the default posture for using the primary layer and is consistent with the wrist orientation when typing on a physical keyboard. The INDEXPOINT-90 and INDEXPOINT-180 gestures are progressively less comfortable postures and the least comfortable posture (INDEXPOINT-180) is allocated to the least frequently used layer. Similarly, the DOUBLEPOINT gesture provides the user with a potential mental association with exerting larger force or being more 'emphatic' and thus provides a logical correspondence with capitalization. Alternative gestures could be used to achieve the same function but our particular choices were based on the goal of offering simple transformations of a typical mid-air typing posture, that is, palm down and with all fingers retracted except the index finger.

3.3 Hand Gesture Recognizer

The recognition system is a single-frame spatial self-attention neural network, directly simplified from Chen et al. [4] by removing the temporal aspect of the model. The input format is a hand skeleton

graph of dimension (J, D), where $J = 4$ is the number of joints that are used for the recognition and $D = 7$ describes the pose (3D position and 4D quaternion) of each joint of interest. We used the wrist, thumb, index and middle fingertips since these were the most relevant joints for distinguishing between our proposed gestures.

The input hand graphs are first linearly mapped to 128-dimensional vectors, which are then passed to a multi-head self-attention layer. The attention layer effectively computes the spatial relationship between each joint of the hand skeleton for that particular gesture. The output is a 128-dimensional attention representation of the gesture, which is passed through a fully connected layer with a softmax function to obtain the final classification results. Unlike previous works on gesture recognition we simplified our model to make single-frame predictions because all supported gestures are static, which means we are not interested in the temporal association between frames. Thus there is no need to provide the model with more than one frame per prediction. The single-frame approach also makes the model much more light-weight and permits faster inference.

4 Evaluation

We evaluate gesture-based layer switching by comparing it in a user study with a button-based baseline method. The baseline method works analogously to the conventional button-based switching mechanism that is commonly used on smartphones and tablets.

The design of the user study is a within-subjects experiment with a single independent variable (the means of switching the keyboard layer) with two levels: BUTTON and GESTURE. The dependent variables are entry rate, keystrokes per character, average error count, layer switching interkey interval, perceived workload (NASA-TLX) and subjective user experience.

4.1 Participants

We recruited a total of 16 participants (mean age = 26.4 $\pm$ 3.2 standard deviation; min=21, max=35; 4 female, 12 male) for the study via convenience sampling. Six participants stated they had no prior experience with VR HMDs. All participants were right-handed. We also asked our participants to report their self-rated speed when typing on a smartphone. The average score was 3.2 $\pm$ 0.9 on a scale from 1 – very slow to 5 – very fast.

4.2 Apparatus and Materials

The study was carried out using a Meta Quest 2 VR HMD and the experimental application was built in Unity. The gesture recognition system was implemented using the Pytorch library and trained with hand gesture data collected from the first two authors of this paper. The training data consists of 20 clips of hand skeleton data from each person for each gesture, and the final model achieves a 99% prediction accuracy on the test set with a 3:1 train-test split. The trained model was then exported to the Open Neural Network Exchange (ONNX) format and integrated into Unity using the Barracuda package[1]. We built the system into an Android Package (apk) that can be deployed onto the HMD and run on-device. However, the actual study was run on a computer via Oculus Link as this provided a more convenient setup for observing participant behavior. The gesture recognizer was configured to run at 10 Hz during the study. Selecting a suitable frequency for running the recognizer involves a trade-off between recognition latency and processor consumption. In our preliminary design work, we found that running the recognizer at faster than 10 Hz delivered no detectable performance advantage.

[1] https://docs.unity3d.com/Packages/com.unity.barracuda@0.7/manual/index.html

4.3 Familiarization Phase

At the beginning of the study, each participant completed a familiarization task that introduced the gestures used in the gesture-based layer-switching method. Participants were required to 'touch' targets that would appear at a random position (but at a fixed depth) while performing the gesture that was displayed on the target. There were a total of 50 target selections in this phase.

4.4 Task

Thereafter the participant was asked to engage in the experimental task in two conditions (BUTTON and GESTURE). The order of the conditions was counterbalanced. In the task the participant entered a set of passwords according to instructions presented in the HMD display. Each password consisted of 12 characters that were randomly generated using an online password generator[2]. Each generated password contained at least one of the lower-case and capitalized letters, numerals and special characters. These generated passwords were therefore compliant with guidance from Google[3] and Microsoft[4], which both recommend a mixture of character types and a minimum length of 12 characters. The task was split into two distinct stages, and the participant was asked to complete both stages in one condition before they proceeded to the alternative condition. At the end of each stage the participant was asked to complete a NASA-TLX questionnaire and provide additional feedback in the form of ratings of speed, accuracy and comfort.

4.4.1 Stage 1

Stage 1 consisted of 20 distinct passwords. Participants were instructed to enter each password as quickly and as accurately as possible. The 20 passwords were split equally into two blocks and participants were given a short break between each block. To control for difficulty of the passwords presented to participants, we compiled two different sets of 20 passwords (40 passwords in total). These password sets were balanced across the participant group both in terms of the two conditions as well as the two possible orders of exposure.

4.4.2 Stage 2

The purpose of Stage 2 was to assess the performance of the two conditions (BUTTON and GESTURE) when the participant's learning has saturated by providing the participant with sufficient practice for learning a single password. The protocol is intended to partially replicate the circumstance in which a user is entering a password that they know and have entered many times before. Participants therefore repeatedly entered the same password 20 times, with these entries again split equally into two blocks. The same password ('5+Roz7Wi$6dR') was used throughout the study for both conditions and for all participants.

5 Results

5.1 Entry Rate

5.1.1 Stage 1: Distinct Passwords

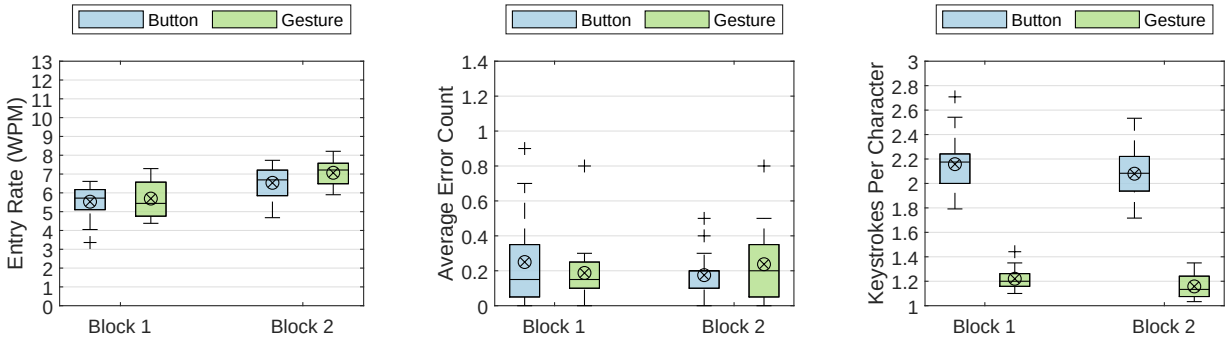
We measure entry rate in words-per-minute (WPM) with a word defined as five consecutive characters including space. The entry rates for the BUTTON and GESTURE conditions are shown in Figure 3a.

In Block 1 (i.e. the first 10 passwords) the mean entry rate was 5.5 WPM for BUTTON and 5.7 WPM for GESTURE. In Block 2 the mean entry rate was 6.5 WPM for BUTTON and 7.1 WPM for GESTURE. Repeated measures analysis of variance at an initial significance level

[2] https://my.norton.com/extspa/passwordmanager?path=pwd-gen

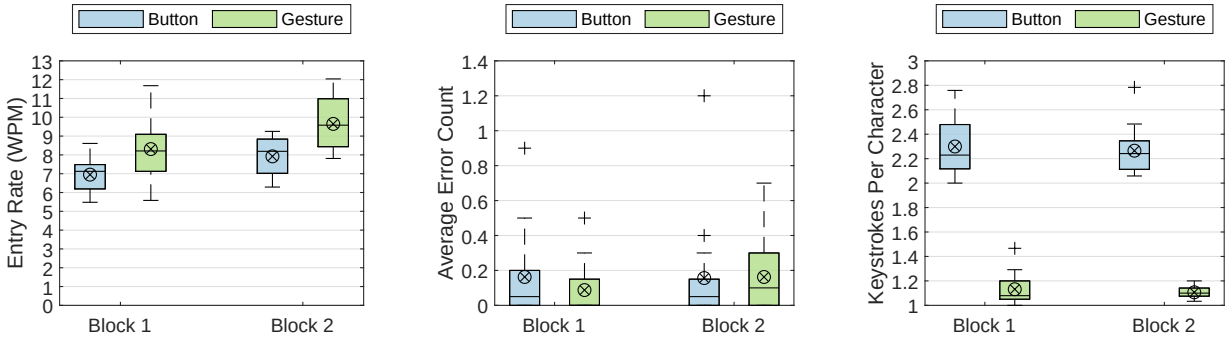
[3] https://support.google.com/accounts/answer/32040?hl=en

[4] https://support.microsoft.com/en-us/windows/create-and-use-strong-passwords-c5cebb49-8c53-4f5e-2bc4-fe357ca048eb



(a) Entry rate in words-per-minute (WPM). (b) Average error count. (c) Keystrokes per character (KSPC).

Figure 3: The entry rate, average error count and keystrokes per character for the distinct password entry task. The boxplots show the median (the horizontal line), the first and third quartile (the box) and the minimum and maximum (the whiskers). The plus signs ('+') indicate outliers. The mean is indicated by the '⊗' symbol.



(a) Entry Rate (WPM). (b) Average Error Count. (c) Keystrokes per Character (KSPC).

Figure 4: The entry rate, average error count and keystrokes per character for the repeated password entry task. The boxplots show the median (the horizontal line), the first and third quartile (the box) and the minimum and maximum (the whiskers). The plus signs ('+') indicate outliers. The mean is indicated by the '⊗' symbol.

of $\alpha = 0.05$ showed that the difference between the two conditions in Block 1 was not significant ($F_{1,15} = 0.180$, $\eta_p^2 = 0.012$, $p = 0.677$) while the difference in Block 2 was significant ($F_{1,15} = 6.755$, $\eta_p^2 = 0.311$, $p = 0.020$). In other words, after some practice, the gesture-based method is significantly faster than the established button-based approach and provides an 8.3% improvement in entry rate.

Figure 3c shows the mean Keystrokes Per Character (KSPC) for both conditions, which reveals the underpinning mechanism explaining the performance improvement. For the BUTTON condition the average KSPC is more than 2 as often at least one keystroke is necessary for layer switching and some special characters require two keystrokes to reach their layer. For the GESTURE condition, the participants can achieve a minimum KSPC of 1 if they execute the task perfectly, thus saving time during text entry.

5.1.2 Stage 2: Repeated Password

The entry rates for the BUTTON and GESTURE conditions are show in Figure 4a. The mean entry rate in Block 1 was 7.0 WPM for BUTTON and 8.3 WPM for GESTURE. In Block 2 the mean entry rates increased further to 7.9 WPM for BUTTON and 9.7 WPM for GESTURE. Repeated measures analysis of variance with an initial significance level of $\alpha = 0.05$ revealed that the differences are statistically significant for both Block 1 ($F_{1,15} = 11.774$, $\eta_p^2 = 0.440$, $p = 0.004$), and Block 2 ($F_{1,15} = 19.197$, $\eta_p^2 = 0.561$, $p = 0.001$). In other words, with more practice participants are able to type even faster with the gesture-based method. As shown in Figure 4c, the mean KSPC for the GESTURE condition approaches unity. This is because participants become more proficient with the technique. In contrast, the KSPC for the BUTTON condition remains above 2.

5.2 Average Error Count

5.2.1 Stage 1: Distinct Passwords

A higher entry rate does not necessarily show the benefit of the gesture-based method if the accuracy of text entry is compromised. Figure 3b plots the average error count during each block of Stage 1, and shows an average error of approximately 0.2 characters per password across all participants for both conditions. This is the equivalent of making a single mistake every fifth password entry, which we consider to be a reasonable level of error. Therefore, the level of error is acceptable in both conditions. A Friedman test revealed no significant difference between the two conditions. This shows that there is no evidence that the gesture-based method would be more error-prone compared to the baseline.

5.2.2 Stage 2: Repeated Password

We observe in Figure 4b that the average error counts for both conditions drop below 0.2 errors per password as the participant makes fewer mistakes at this stage of the study. A Friedman test revealed no significant difference between the two conditions.

Table 1: Median and interquartile range (IQR) of the questionnaire responses to Questions 1 to 3. Responses were recorded on a five point Likert scale from 1—strongly disagree to 5—strongly agree.

		BUTTON		GESTURE	
Statement		MEDIAN	IQR	MEDIAN	IQR
Q1	The technique made it easy to switch layers quickly.	3	1	5	1
Q2	The technique made it easy to switch layers accurately.	3.5	1	4	0.25
Q3	The technique was comfortable to use.	3	2	4	0.25

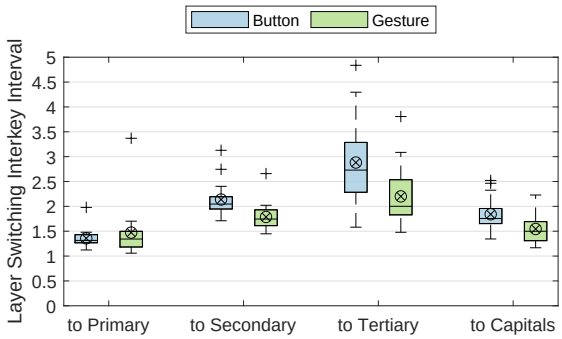


Figure 5: Boxplot of the mean interkey interval for key presses involving a layer switch, grouped according to the target keyboard layer. The mean is indicated by the '⊗' symbol.

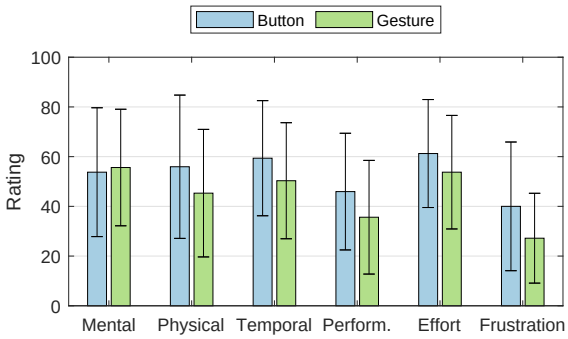


Figure 6: Mean NASA-TLX scale ratings. Error bars show ± 1 standard deviation. Note that a lower 'Performance' rating indicates 'better' perceived performance.

5.3 Layer Switching Interkey Interval

We also investigate the average time for consecutive key presses that require a layer transition. This allows us to obtain a more detailed understanding of where the performance advantage of the GESTURE condition is realized. The time between consecutive key presses is also referred to as the interkey interval (IKI). Here we define layer switching IKI as the time between two consecutive key presses on the main keyboard layout that result in a correct character selection and where these two characters are on different layers. Note that in the BUTTON condition there may be one or more intervening presses of the mode switch buttons which are then ignored when computing the layer switching IKI. We computed the average layer switching IKI separately for each of the four possible layers to which the user is transitioning. To ensure a representative amount of samples within each layer group we combined the data from all blocks and stages.

Figure 5 shows a boxplot of the participant mean layer switching IKI in both conditions and grouped by target layer on the x-axis. For the BUTTON condition, the average IKI for a layer transition was 1.36 s (primary layer), 2.14 s (secondary layer), 2.88 s (tertiary layer), and 1.84 s (capitalization layer). For the GESTURE condition, the average IKI for a layer transition was 1.47s (primary layer), 1.79s (secondary layer), 2.20s (tertiary layer), and 1.54s (capitalization layer). The layer switching IKI was shorter in the GESTURE condition for all target layers except the default layer. A repeated measures analysis of variance revealed that this reduction in layer switching IKI was significant for the secondary layer ($F_{1,15} = 10.244$, $\eta_p^2 = 0.406$, $p = 0.006$), tertiary layer ($F_{1,15} = 14.126$, $\eta_p^2 = 0.485$, $p = 0.002$) and capitalization layer ($F_{1,15} = 7.226$, $\eta_p^2 = 0.325$, $p = 0.017$). The difference was not significant for the primary (lower-case alpha) layer. The negligible difference observed for switches to the primary layer can be explained by the design of the BUTTON condition. The BUTTON condition always allows returning to the primary layer with a single key press regardless of which layer is before the transition. The reduction in switching time for all other target layers, together with the KSPC results, helps to explain the mechanism for the entry rate increase observed for our gesture-based layer switching method.

5.4 Perceived Workload

Figure 6 plots the mean raw NASA-TLX scale scores across all participants. The GESTURE condition was considered by participants to be less physically and temporally demanding, likely due to the fewer keystrokes required and the higher entry rate. Participants also perceived themselves to perform better, experience lower frustration, and use less effort in the GESTURE condition. The gesture method introduced slightly higher mental load, as users were required to learn and adapt to a completely new text entry technique within a relatively short period, whereas the button-based condition required very little adaptation. This point was also reflected in qualitative feedback from participants. A Friedman test revealed that none of these measured differences were statistically significant.

5.5 Participant Feedback

Participants completed a final written questionnaire which included a specific question eliciting their preference between the two conditions. 15 out of the 16 participants indicated that they preferred the gesture-based method overall. Participants were also asked to rate each technique by answering the three questions shown in Table 1. The results show that participants regarded the gesture method as faster, slightly more accurate and more comfortable to use. A Friedman test revealed significant differences for Q1 ($p < 0.001$), but not for Q2 ($p = 0.206$) or Q3 ($p = 0.0578$). For BUTTON condition, 11 participants commented that it was 'familiar'. Nine participants said they had to press multiple times to reach a wanted character and two participants found this condition to be more error-prone. Meanwhile, seven participants expressed that the gesture method was 'fast' or 'efficient' while seven participants described the method using the words 'easy', 'natural' or 'convenient'. However, four participants expressed that the some gestures (DOUBLEPOINT, INDEXPOINT-180) were less comfortable to use, and four participants mentioned 'false activation' and 'inaccurate hand tracking'. Although infrequent, these tracking issues were most commonly observed to occur when the native Quest 2 hand-tracking falsely reported the middle finger as being extended when it was in fact retracted or vice versa.

6 Discussion

The user study reveals that the gesture-based mode switching method achieves a higher entry rate by reducing the KSPC close to a minimum and does not introduce additional errors. Furthermore, the gesture-based method helps to reduce the time between consecutive key presses that fall on different layers. After some practice, users obtain an 8.3% increase in entry rate with the gesture-based method on distinct and unfamiliar passwords. With additional practice on a single password, the performance increased by a significant 21.8% compared to the baseline. Further, users were able to learn the new technique over a relatively short period of time and were thereafter able to outperform the traditional button-based method. Despite no prior familiarity, the gesture-based method was preferred by almost all (15 out of 16) participants in the user study.

We observed occasional unwanted keyboard layer selections when using the gesture-based method, which resulted in input errors. There are three primary potential sources of errors causing this: 1) the participant performed an incorrect gesture; 2) the HMD hand tracking was inaccurate; and 3) misrecognition by the recognizer. The recognizer used in the study was trained on data from two people. More training data from a wider pool of users would reduce the risk of over-fitting and potentially improve recognition accuracy.

7 Limitations and Future Work

We believe that our work examines a well-motivated operating point in the design space for gesture-based keyboard layer switching but we acknowledge that there are other operating points that can be explored in the future. Other gesture choices, including the use of bimanual gestures, could potentially be more comfortable or intuitive for users and deliver better performance. We leave the examination of the possible merits of alternative gesture sets as future work.

The efficacy of gesture-based keyboard layer switching is clearly demonstrated by the results we obtained from our controlled user study. Nevertheless, we recognize that our experiment necessarily relies on a task formulation that is somewhat contrived and that our participant pool is relatively small. Further studies are necessary to understand how the technique may perform with a larger and more diverse user group, both in controlled and in-the-wild settings. A more detailed examination of the causes of errors in layer switching would also be informative. However, this is difficult to perform without interrupting participants during execution since intentional switching and false activations are difficult to differentiate based on logged data alone.

We also recognize that although we have examined the use case of password entry, we have not considered broader security concerns. Prior work (e.g. [19]) has studied the potential security concerns due to shoulder-surfing when entering passwords on an immersive HMD and proposed potential solutions to this form of attack. We leave further such refinements as future work.

8 Conclusions

In this paper we have introduced a new technique for keyboard layer-switching based on hand gestures in order to support a more efficient method of special character entry. The method allows participants to seamlessly switch between different layers on a virtual keyboard without interrupting their typing.

Our evaluation with 16 participants typing passwords compared the new gesture-based method with a commonly used approach that uses dedicated buttons to switch between layers. This evaluation showed that the gesture-based method allows users to type special characters faster with the same level of accuracy. With practice, participants achieved a significant 8.3% increase in entry rate compared to the baseline and with additional practice with a single password the increase in entry rate grew to a significant 21.8%. In addition, 15 out of 16 participants preferred the gesture-based method. Finally, we also note that the gesture-based method can be integrated such that it is compatible with the established button-based mechanism for switching layers. This could provide an optional complementary entry method for expert users to efficiently enter special characters. Given this, in combination with the encouraging empirical results, we believe the gesture-based method can be a popular way of supporting special character entry on virtual keyboards in VR.

Acknowledgments

John Dudley and Per Ola Kristensson were supported by the EPSRC (grant EP/S027432/1). Supporting data is available at `https://doi.org/10.17863/CAM.87514`.

References

[1] J. Adhikary and K. Vertanen. Text Entry in Virtual Environments using Speech and a Midair Keyboard. *IEEE Transactions on Visualization and Computer Graphics*, 27(5):2648–2658, 2021. doi: 10.1109/TVCG.2021.3067776.

[2] S. K. Card, T. P. Moran, and A. Newell. The Keystroke-Level Model for User Performance Time with Interactive Systems. *Communications of the ACM*, 23(7):396–410, Jul. 1980. doi: 10.1145/358886.358895.

[3] D. L. Chen, R. Balakrishnan, and T. Grossman. Disambiguation Techniques for Freehand Object Manipulations in Virtual Reality. In *2020 IEEE Conference on Virtual Reality and 3D User Interfaces (VR)*, pages 285–292, 2020. doi: 10.1109/VR46266.2020.00048.

[4] Y. Chen, L. Zhao, X. Peng, J. Yuan, and D. N. Metaxas. Construct Dynamic Graphs for Hand Gesture Recognition via Spatial-Temporal Attention. In *30th British Machine Vision Conference*, 2019. doi: 10.48550/ARXIV.1907.08871.

[5] T. J. Dube and A. S. Arif. Text Entry in Virtual Reality: A Comprehensive Review of the Literature. In M. Kurosu, editor, *Human-Computer Interaction. Recognition and Interaction Technologies*, Lecture Notes in Computer Science, pages 419–437, Cham, 2019. Springer International Publishing. doi: 10.1007/978-3-030-22643-5_33.

[6] J. J. Dudley, K. Vertanen, and P. O. Kristensson. Fast and Precise Touch-Based Text Entry for Head-Mounted Augmented Reality with Variable Occlusion. *ACM Transactions on Computer-Human Interaction*, 25(6):30:1–30:40, Dec. 2018. doi: 10.1145/3232163.

[7] J. J. Dudley, H. Benko, D. Wigdor, and P. O. 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. doi: 10.1109/ISMAR.2019.00027.

[8] P. Evtimova and J. Nicholson. Exploring the Acceptability of Graphical Passwords for People with Dyslexia. In *Human-Computer Interaction – INTERACT 2021*, page 213–222, Berlin, Heidelberg, 2021. Springer-Verlag. doi: 10.1007/978-3-030-85623-6_14.

[9] C. R. Foy, J. J. Dudley, A. Gupta, H. Benko, and P. O. Kristensson. Understanding, Detecting and Mitigating the Effects of Coactivations in Ten-Finger Mid-Air Typing in Virtual Reality. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*, CHI '21, New York, NY, USA, 2021. ACM. doi: 10.1145/3411764.3445671.

[10] M. Frutos-Pascual, C. Gale, J. M. Harrison, C. Creed, and I. Williams. Character Input in Augmented Reality: An Evaluation of Keyboard Position and Interaction Visualisation for Head-Mounted Displays. In *Human-Computer Interaction – INTERACT 2021*, page 480–501, Berlin, Heidelberg, 2021. Springer-Verlag. doi: 10.1007/978-3-030-85623-6_29.

[11] C. George, M. Khamis, D. Buschek, and H. Hussmann. Investigating the Third Dimension for Authentication in Immersive Virtual Reality and in the Real World. In *2019 IEEE Conference on Virtual Reality and 3D User Interfaces (VR)*, pages 277–285, 2019. doi: 10.1109/VR.2019.8797862.

[12] A. Gupta, C. Ji, H.-S. Yeo, A. Quigley, and D. Vogel. RotoSwype: Word-Gesture Typing using a Ring. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, CHI'19, pages 1–12. ACM, New York, NY, USA, May 2019.

[13] A. Gupta, M. Samad, K. Kin, P. O. Kristensson, and H. Benko. Investigating Remote Tactile Feedback for Mid-Air Text-Entry in Virtual Reality. In *2020 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, pages 350–360, 2020. doi: 10.1109/ISMAR50242.2020.00062.

[14] P. O. Kristensson. Five Challenges for Intelligent Text Entry Methods. *AI Magazine*, 30(4):85–85, 2009.

[15] X. Lu, D. Yu, H.-N. Liang, W. Xu, Y. Chen, X. Li, and K. Hasan. Exploration of Hands-free Text Entry Techniques for Virtual Reality. In *2020 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, pages 344–349, 2020. doi: 10.1109/ISMAR50242.2020.00061.

[16] M. McGill, S. Brewster, D. P. De Sa Medeiros, S. Bovet, M. Gutierrez, and A. Kehoe. Creating and Augmenting Keyboards for Extended Reality with the Keyboard Augmentation Toolkit. *ACM Transactions on Computer-Human Interaction*, 29(2), Jan. 2022. doi: 10.1145/3490495.

[17] G. B. Mo, J. J. Dudley, and P. O. Kristensson. Gesture Knitter: A Hand Gesture Design Tool for Head-Mounted Mixed Reality Applications. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*, CHI '21, New York, NY, USA, 2021. ACM. doi: 10.1145/3411764.3445766.

[18] K. Pfeuffer, L. Mecke, S. Delgado Rodriguez, M. Hassib, H. Maier, and F. Alt. Empirical Evaluation of Gaze-Enhanced Menus in Virtual Reality. In *26th ACM Symposium on Virtual Reality Software and Technology*, VRST '20, New York, NY, USA, 2020. ACM. doi: 10.1145/3385956.3418962.

[19] D. Schneider, A. Otte, T. Gesslein, P. Gagel, B. Kuth, M. S. Damlakhi, O. Dietz, E. Ofek, M. Pahud, P. O. Kristensson, J. Müller, and J. Grubert. ReconViguRation: Reconfiguring Physical Keyboards in Virtual Reality. *IEEE Transactions on Visualization and Computer Graphics*, 25(11):3190–3201, 2019. doi: 10.1109/TVCG.2019.2932239.

[20] J. Shen, J. Dudley, and P. Kristensson. The Imaginative Generative Adversarial Network: Automatic Data Augmentation for Dynamic Skeleton-Based Hand Gesture and Human Action Recognition. In *2021 16th IEEE International Conference on Automatic Face and Gesture Recognition (FG 2021)*, pages 1–8, Los Alamitos, CA, USA, Dec. 2021. IEEE Computer Society. doi: 10.1109/FG52635.2021.9666999.

[21] R. Shi, N. Zhu, H.-N. Liang, and S. Zhao. Exploring Head-based Mode-Switching in Virtual Reality. In *2021 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, pages 118–127, 2021. doi: 10.1109/ISMAR52148.2021.00026.

[22] M. Speicher, A. M. Feit, P. Ziegler, and A. Krüger. Selection-Based Text Entry in Virtual Reality. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*, CHI '18, page 1–13, New York, NY, USA, 2018. ACM. doi: 10.1145/3173574.3174221.

[23] H. B. Surale, F. Matulic, and D. Vogel. Experimental Analysis of Barehand Mid-Air Mode-Switching Techniques in Virtual Reality. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, CHI '19, page 1–14, New York, NY, USA, 2019. ACM. doi: 10.1145/3290605.3300426.

[24] C. Wang, W. Chu, P. Chiu, M. Hsiu, Y. Chiang, and M. Y. Chen. Palmtype: Using Palms as Keyboards for Smart Glasses. In *Proceedings of the 17th International Conference on Human-Computer Interaction with Mobile Devices and Services*, MobileHCI '15, page 153–160. ACM, 2015. doi: 10.1145/2785830.2785886.

[25] D. Weir, H. Pohl, S. Rogers, K. Vertanen, and P. O. Kristensson. Uncertain Text Entry on Mobile Devices. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '14, page 2307–2316, New York, NY, USA, 2014. ACM. doi: 10.1145/2556288.2557412.

[26] C. Yu, Y. Gu, Z. Yang, X. Yi, H. Luo, and Y. 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*, CHI '17, pages 4479–4488, New York, NY, USA, 2017. ACM. doi: 10.1145/3025453.3025964.